6 入出力装置をOSがどう扱えば良い

6–1 入出力機器を管理する上での課題

次の **2つの課題** がある。

- 入出力機器の制御
- 論理的なインターフェースの提供

p.15からの引用：

入出力機器は機器毎に制御の仕方が違う。

⇨ 入出力などの共通機能を容易に実行できる環境をハードウェアに付随して提供すれば、プログラマの生産性向上につながる。これが初期のOSの目的であった。

入出力機器の制御：

- 入出力機器は多種多様に存在し各々に制御の仕方が異なっているので、それらの制御プログラムをサポートすることは多くの労力を取られる。また、
- ハードウェアの詳細な知識も必要になるため、一般に繁雑で難易度も高い。
- OSはハードウェアの能力を最大限に引き出さなければならない。

例6. 1 (論理的なインターフェースの提供)

UNIX 等の標準的な環境の下でプログラムを作る場合、OS は入出力機器を操作するための論理的なインターフェースをプログラマに提供してくれている。

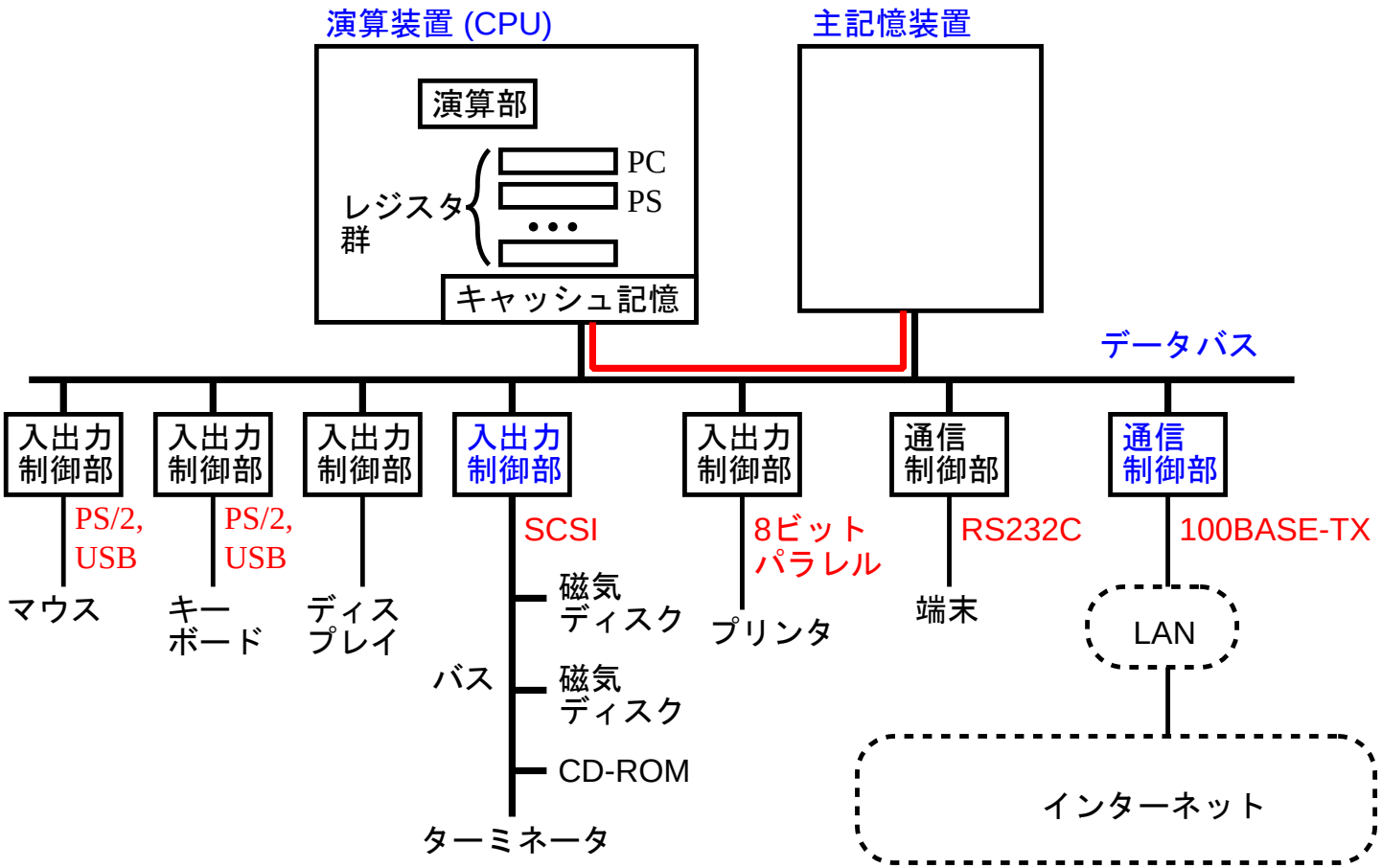
より具体的には、

- プログラマは、
  各々の入出力機器をファイルの一種と見ることができる。
  (操作機器に依存しない統一的なインターフェース。)

  ⇨ 入出力機器が変わってもプログラムの変更は
  ほとんど必要ない。(ファイル名だけ)

- 物理的なインターフェースの実装は全て OS が行う。

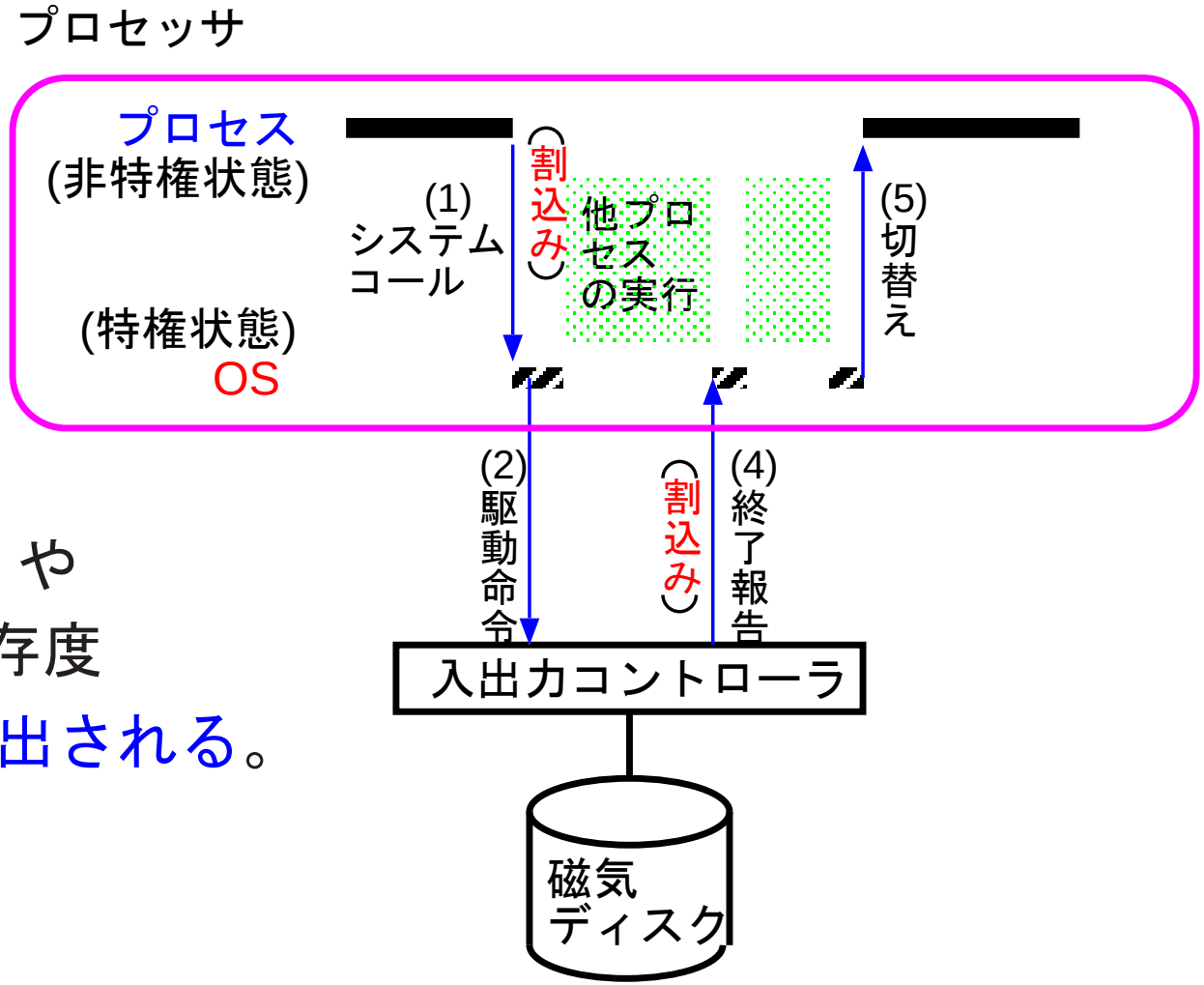
6–2 入出力装置の接続例



- 演算装置と主記憶の間のデータバスは、頻繁なデータ転送を必要としているため、特別なデータバスを用意していることが多い。
- 入出力機器は、商用的な意味から標準インターフェースを備えているものが多い。

6–3 入出力管理・制御のソフトウェア階層

「ファイル読み込み」の大雑把な様子



OSの内部では

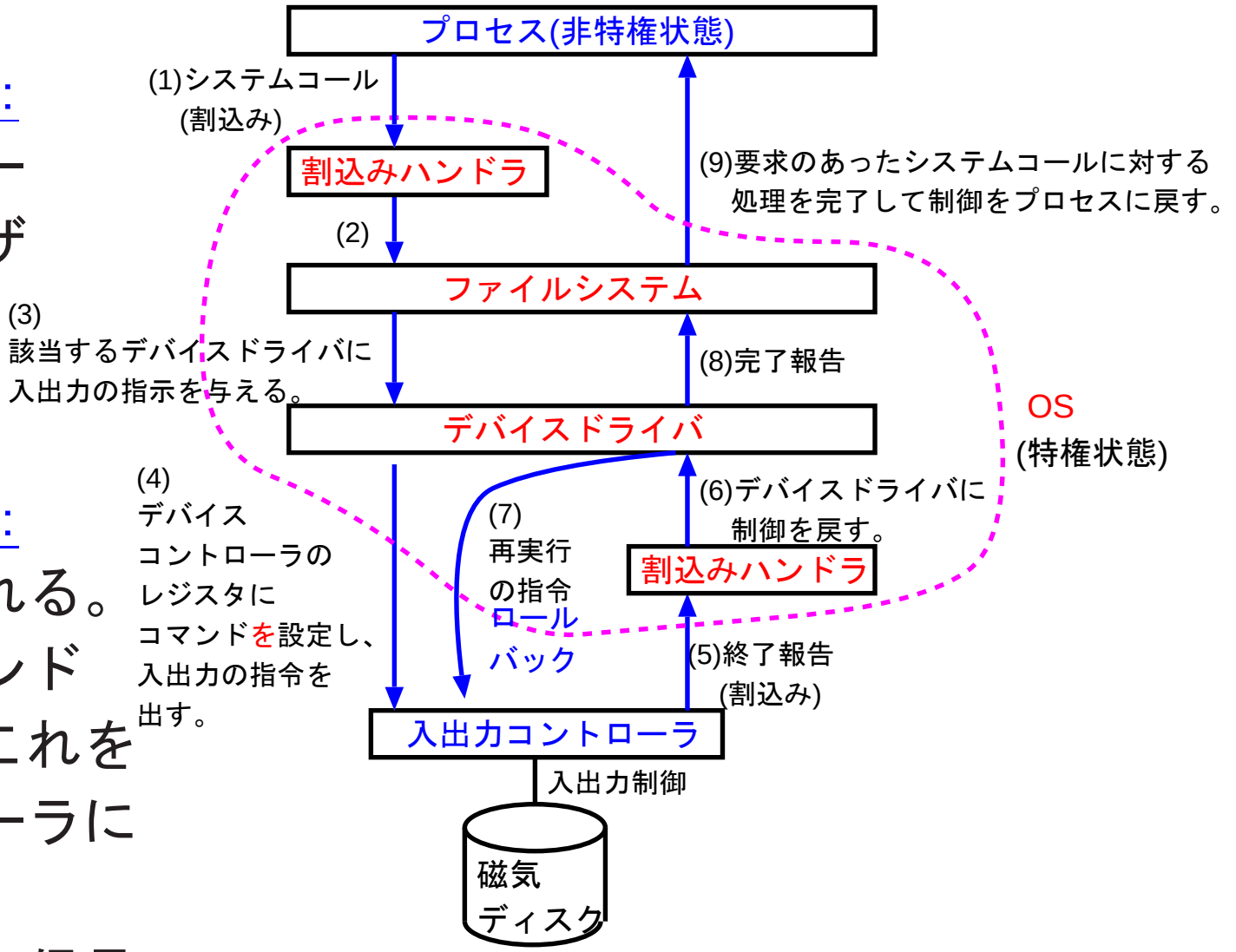
次の様なソフトウェアが

扱うデータの抽象度　や
処理の抽象度/装置依存度

等に基づいて階層的に呼び出される。

- ファイルシステム :
- デバイスドライバ :
- 割込みハンドラ (割込み処理ルーチン) :

- ファイルシステム：
  論理的なインターフェースをユーザに提供するための機能ブロック



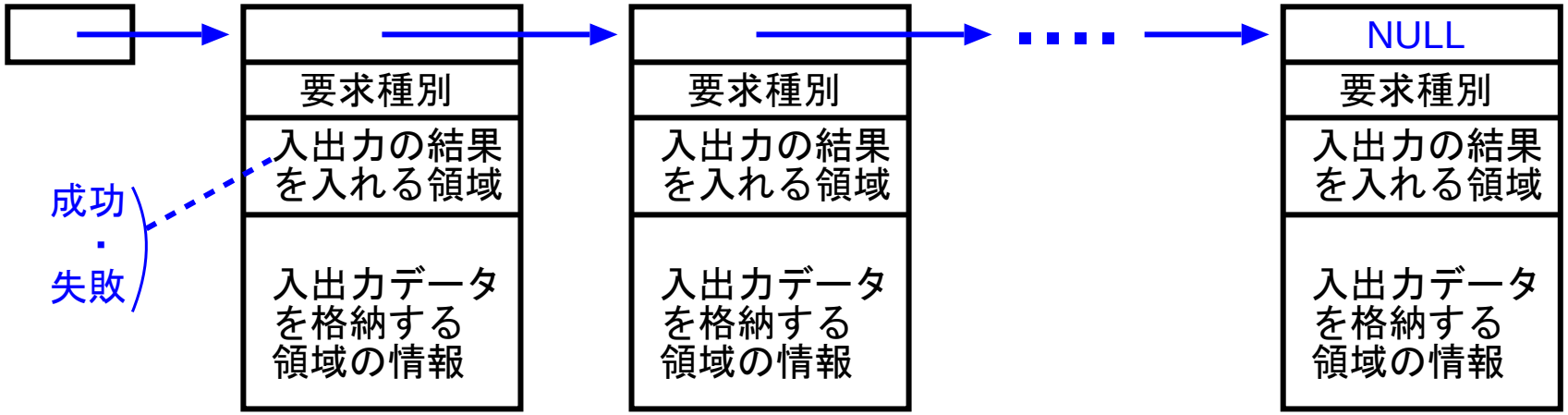
- デバイスドライバ：
  装置毎に用意される。
  装置固有のコマンドの列を生成し、これを入出力コントローラに処理させる。
  終ったら、割込み信号。

- 割込みハンドラ (割込み処理ルーチン)：　起こった割込みに対する処置が書かれたプログラムで、割込みの種類毎に用意されている。

デバイスドライバにおける処理は少々複雑で、次の様になっている。

デバイスドライバ層における入出力機器の起動制御：

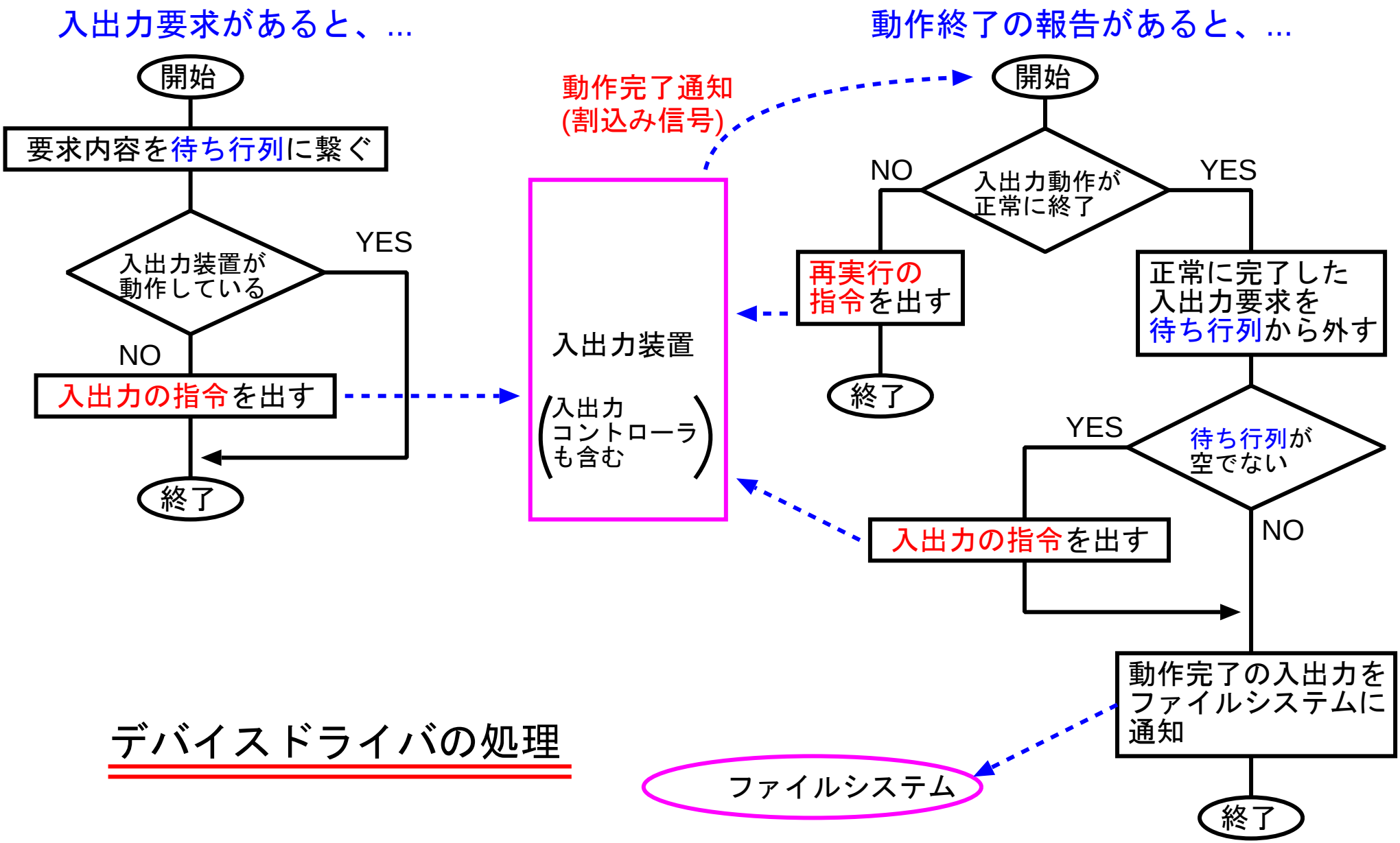
- 入出力要求は入出力装置毎に待ち行列で管理する。



待ち行列の管理についての補足：

優先度の高い入出力要求が発生した場合にそれを待ち行列の先頭に入れるなどして、入出力のスケジューリングを行うこともある。

例えば、磁気ディスクなどでは、...



デバイスドライバの処理

- 入出力要求 がカーネルに対して出されると、CPUの状態が特権状態に切替えられた上でデバイスドライバが呼び出される。そこでは

  (1) その要求が待ち行列に繋がれ、

  (2) 入出力装置に対して処理の指令が出ていなければ指令が出される。

- 入出力装置から「入出力終了」の割り込み通知 がCPUに対して出されると、CPUの状態が特権状態に切替えられ、その装置を担当するデバイスドライバに制御が戻って来る。 そこでは、

  (1) 入出力動作が正常に行われたかどうかがチェックされ、もし正常に行われていなかったらその入出力処理が再実行(ロールバックと言う)される。

  (2) 入出力動作が正常に行われていたならば、
  - ◇ 正常に終了した入出力要求を待ち行列から外す。
  - ◇ この装置の待ち行列を調べ、その先頭の入出力要求を(もしあれば)実行する。一方では、
  - ◇ 入出力動作の完了した入出力要求をファイルシステムに通知する。

6–4 入出力制御の特徴

全般的な特徴：

- 割込み制御で実現されている部分が多い。

  その理由：

  ◇プロセッサ以外の装置からの通信は割込み信号。

  ◇代替案として例えば入出力の処理をプロセス化すると、それらのプロセスの起動・停止などの処理が増えてシステム全体として運用が非効率になる。

- 障害への処理を重視。
- 装置の種類に合わせ多種類の制御法を実現。

  例えば：

  磁気ディスク装置に対しては複数のプロセスから同時に入出力の要求が発生することがある。これらの要求を効率的に処理できないといけない。

外部記憶装置制御の特徴：

装置との間のコマンドの送受が処理の大半を占めるが、現在ではこのコマンド送受の処理は次の2つの理由で簡単になっている。

- 装置の高機能化。

  ⇨ 装置内部の構造を意識せずにデータの送受信が可能。
  コマンドの簡略化。

- 入出力制御のハードウェア化。
  (コマンドの送受を制御するLSIが登場。)

印刷装置制御の特徴：

- 排他制御を行う必要があるので、少し複雑になる。
  (出力が混ざらない様に)

- データの形式が装置の種類毎に異なることが多い。

  ⇨ データの形式はライブラリで実現されていることが多い。

- 用紙切れ、トナー切れの場合はCPUに知らせる。

6–5 基礎知識 磁気ディスクに関する基本的な用

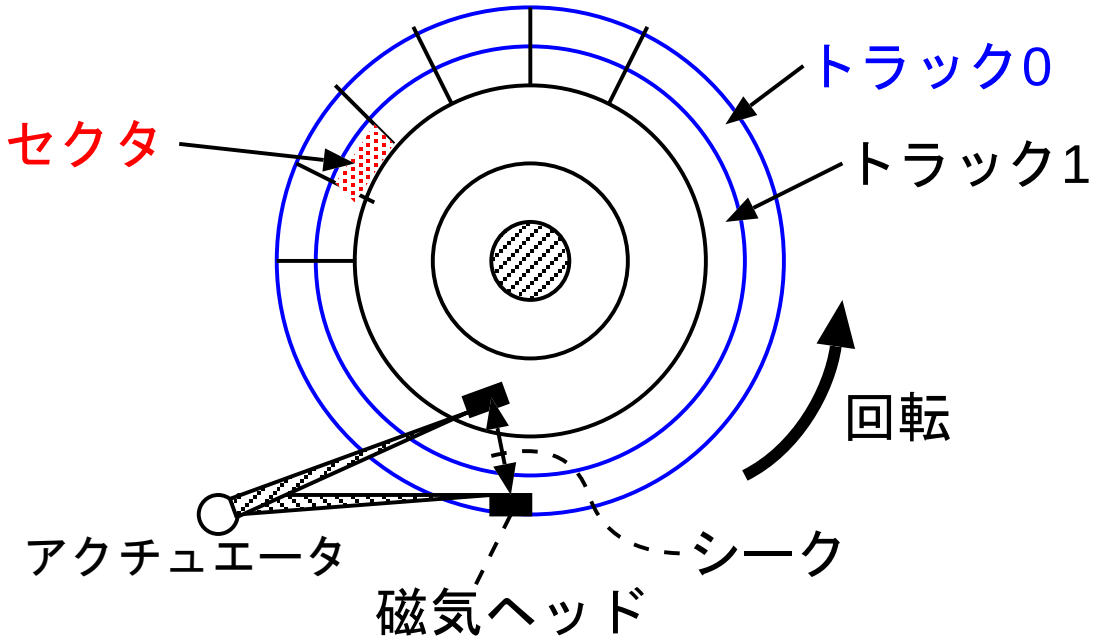
磁気ディスク：

- 14インチ径, 9インチ径, 8インチ径, 5.25インチ径, 3.5インチ径(PCデスクトップ型用), 2.5インチ径(Notebook型用)のものがあり、容量が年々増大している。

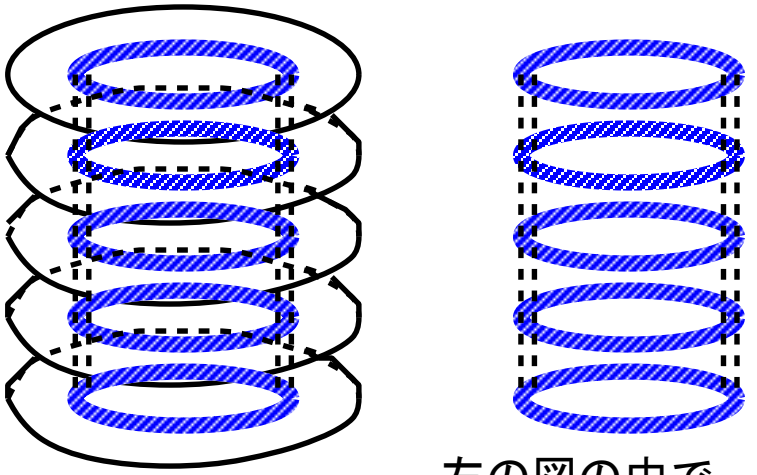
ちなみに、情報工学科実習室のディスクアレイの容量は
　1枚当たり ??GB、
　総計で 10TB 以上
である。

磁気ディスクの構造：

- アルミニウムやガラスの円盤上に磁気膜を塗布し情報を記録する。



- 磁気ディスクの回転に伴って磁気ヘッドが走査する同心円上の部分をトラックと言う。
- トラックは記録の単位であるが、一般にはトラックを更に128~1024バイトの固定長のセクタという単位に細分して使用される。
- 円盤が複数重ねられた場合は、同一径のトラックが上下に円盤の枚数(×2?)だけ存在することになる。これらのトラックの集まりをシリンダと呼ぶ。



左の図の中で、
これらの部分がシリンダ

cylinder n. 円筒,(幾) 円柱

同一シリンダ内の情報へのアクセスは磁気ヘッドを動かす必要がなく電子的なスイッチの切替えだけで行うことが出来るので、1つの記録の単位として扱われる。

磁気ディスクの動作：

磁気ディスク上の情報へのアクセスは、次の様に行われる。

(1) アクチュエータが機械的に動作して磁気ヘッドを目的とするトラックまで移動させる。 — シーク時間

(2) 使用する磁気ヘッドの切替え。

(3) 目的のセクタが磁気ヘッドの下まで来るのを待つ。 — サーチ時間

(4) 情報を転送する。 — 転送時間

⇨ アクセス時間≈シーク時間＋サーチ時間＋転送時間

補足： アクセス時間を実測した結果によれば、

- シーク時間 ··· 全体の60%
- サーチ時間 ··· 全体の32%
- 転送時間 ··· 全体の8%

⇨ このような分析結果に基づいて、磁気ディスク上の情報へのアクセス要求に対するサービスのスケジューリングが工夫される。

6–6 周辺入出力制御の手法(1)
—データの基本単位の違いを処理する—

補足：
CPUが直接アクセスできるのは主記憶内のデータだけ。

⇨ 外部記憶内のデータを使うには、一旦
主記憶 ⟷ 外部記憶の間のデータ受渡しが必要。

外部記憶は(主記憶に比べて)大容量なので、
ある一定の大きさのデータの塊を処理の最小単位としている。
(ここではブロックと呼ぶ;磁気ディスクの場合はセクタ)

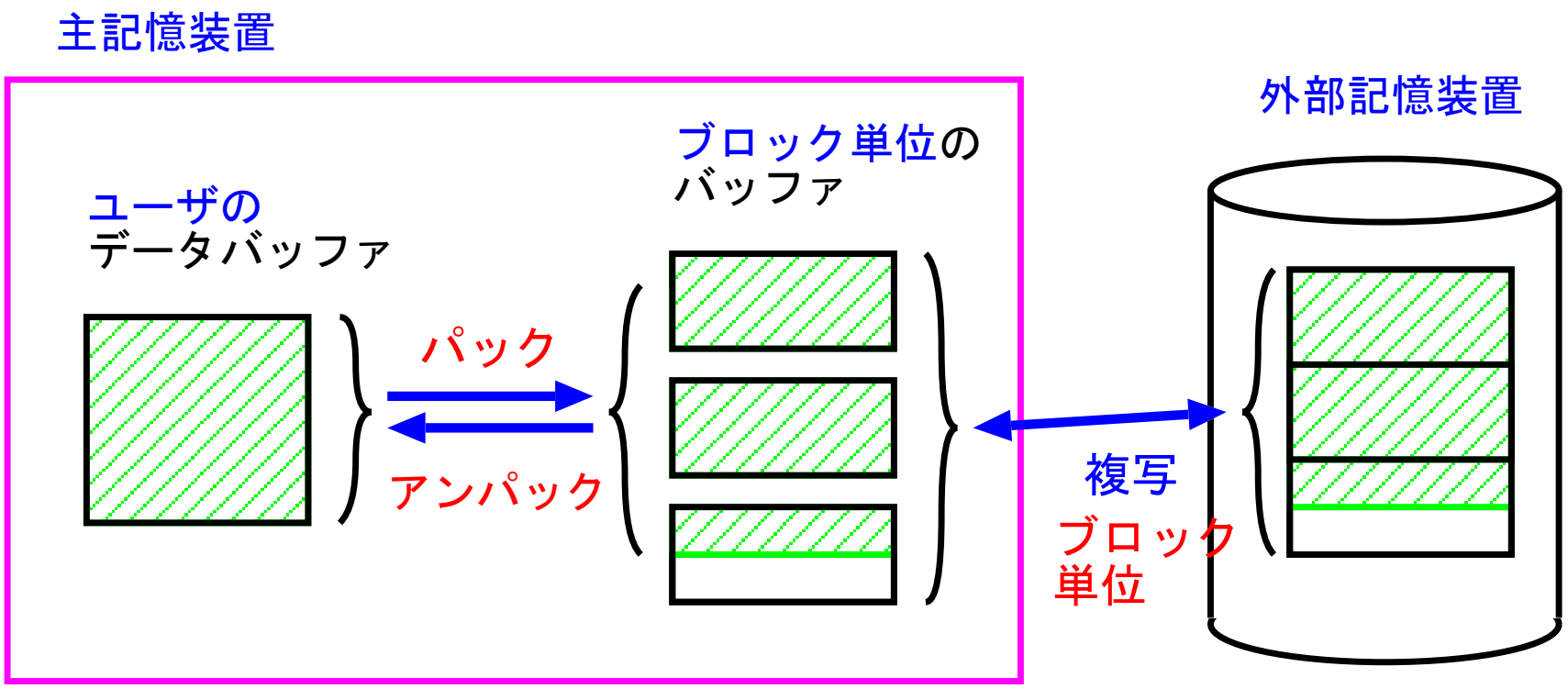
⇨ 主記憶と外部記憶の間でデータの受渡しを行う際には、主記憶内のデータをブロック単位に揃えたりbyte単位に戻したりしなければならない。

—

⇨ 主記憶と外部記憶の間でデータの受渡しを行う際には、主記憶内のデータをブロック単位に揃えたりbyte単位に戻したりしなければならない。

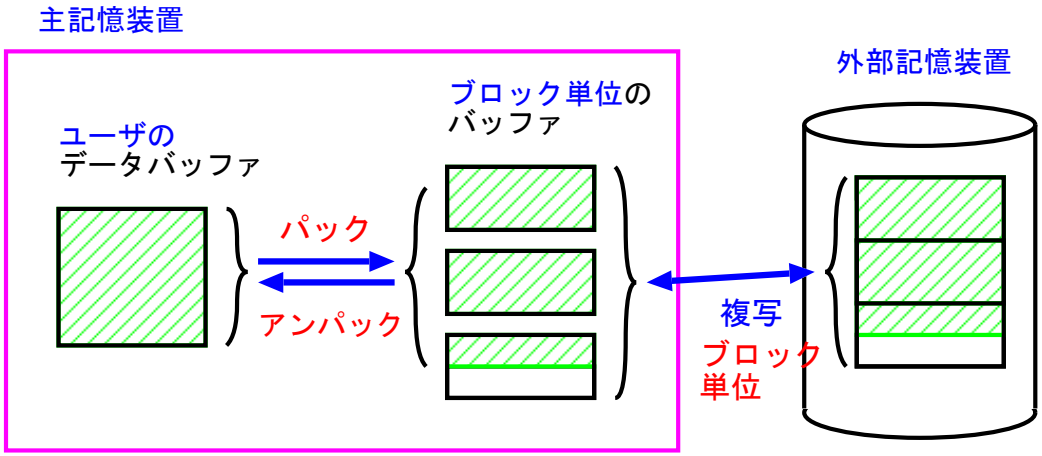
パック ： byte単位 $\Longrightarrow$ ブロック単位

アンパック ： ブロック単位 $\Longrightarrow$ byte単位



外部記憶への出力：

(1) 出力したいブロックの位置を決定する。
(2) ブロック単位のバッファを必要な個数確保する。
(3) ユーザのデータバッファからブロック単位のバッファにデータを複写する。 (パック操作)
(4) ブロック単位のバッファから外部記憶へ出力。



外部記憶からの入力：

(1) 入力したいブロックの位置を決定する。
(2) ブロック単位のバッファを必要な個数確保する。
(3) 外部記憶からブロック単位のバッファへ入力。
(4) ブロック単位のバッファからユーザのデータバッファにデータを複写する。 (アンパック操作)

6–7 周辺入出力制御の手法(2) —高速化—

入出力を効率的に行うための手法として次のようなものがある。

データのキャッシュ化：

外部記憶のブロックを置くためのキャッシュ領域を主記憶上に用意し、そこに外部記憶上の頻繁にアクセスするブロックを保持する。

(⟹ 1.1節)

先読みと遅延書き出し：

前述のキャッシュ領域を利用する際の工夫。

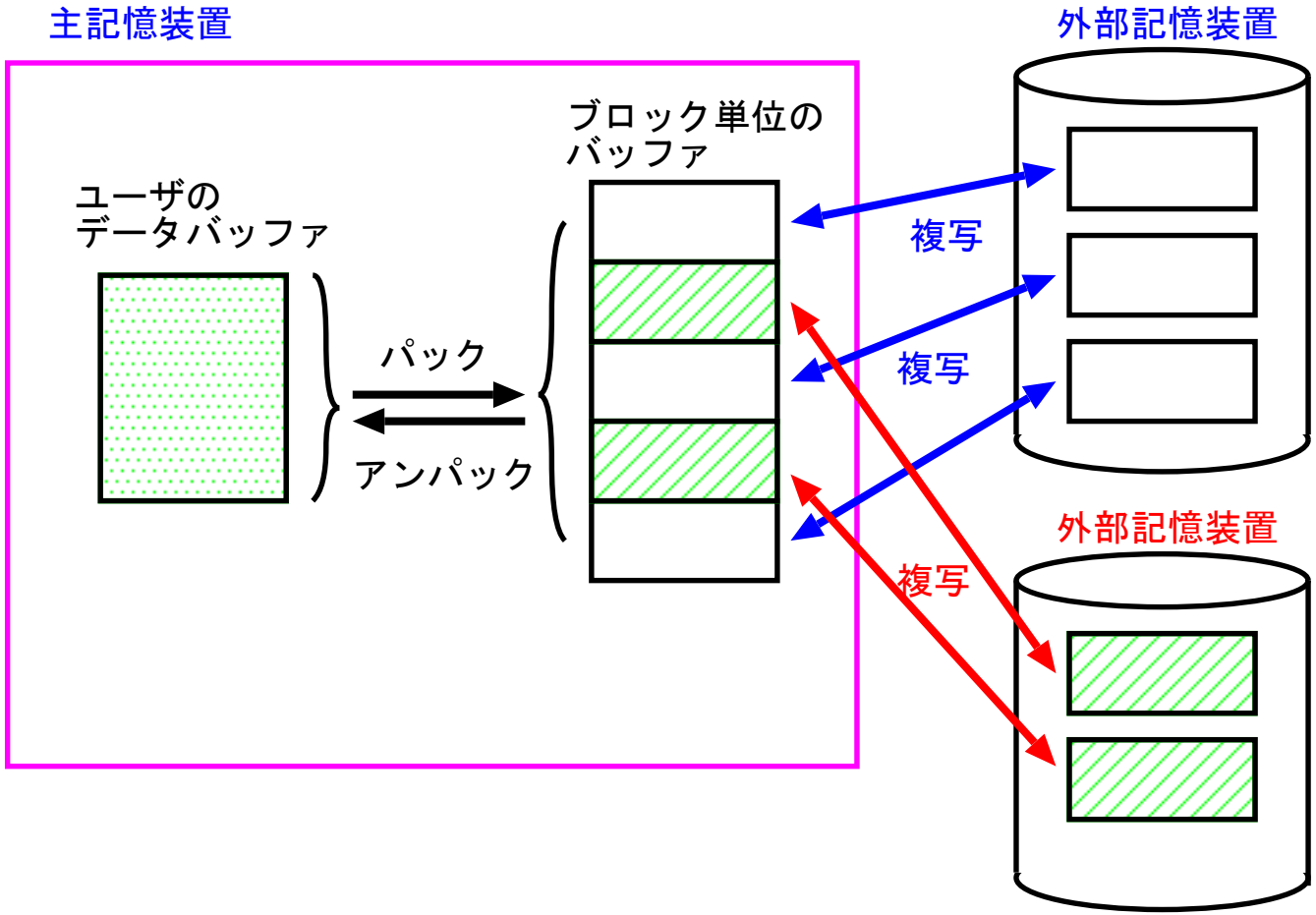
- 先読み··· 外部記憶からのブロックの読み込みの際、次のブロックも一緒に読み込んでしまう。
  (プログラムの局所参照性を仮定。)

- 遅延書き出し··· キャッシュ領域に書き込みがあっても、その結果をすぐに外部記憶に複写することはしない。キャッシュ領域を空ける必要が出た時点で始めて外部記憶に書き込む。

インターリーブ(interleave)処理：

並列に動作する複数の外部記憶装置がある場合、1つの情報(ファイル) をこれらの外部記憶に分散して配置・格納する。

⇨ 異なる外部記憶に配置されたブロックは並列に入出力できる。



アクセス順序の制御：

例えば磁気ディスク装置の場合は入出力要求の待ち行列を管理する方法として次のような方法が考えられている。

- 先着順法 · · · 入出力要求のあった順に待ち行列に繋ぎ処理していく。

- 最短位置法 · · · 各時点で、次の磁気ヘッドの動作が最小になるように次に処理する入出力要求を選ぶ。

  ⇨ 要求した順に処理されないので
  管理しているデータに信頼性がそれほど無い。
  長時間待たされる入出力要求が出る可能性がある。

- スキャン法 · · · 磁気ヘッドを軸方向、外方向に交互に動作させることにし、その流れの中で次に入出力可能な要求を順次捌いていく。

  ⇨ 要求した順に処理されないので
  管理しているデータに信頼性がそれほど無い。

6–8 通信制御の特徴

基本的な特徴：

- 高機能な通信制御用LSIがある。
- データ送受信の前後に通信パスの設定や解放の処理が必要。
- データ受信の処理を高速に行う必要がある。

> 補足：
> 特に高速な通信路の場合は注意。
> データを受信し損なうとデータの再送が行われる。

- データの送受信がうまくできない事態もよくある。
- データ受信の終了が予見できない。

⇨ 長時間データが届いていない場合はデータ受信を取り消す機能も必要。