コンピュータの仕組み
(3)

柴山　潔

コンピュータの仕組み

1　コンピュータシステム
2　ハードウェア
3　内部装置
4　プロセッサ
5　メモリ
6　外部装置
7　論理回路
8　オペレーティングシステム

3 内部装置

3.1 コンピュータの内部装置

3.2 プロセッサとメインメモリとの通信

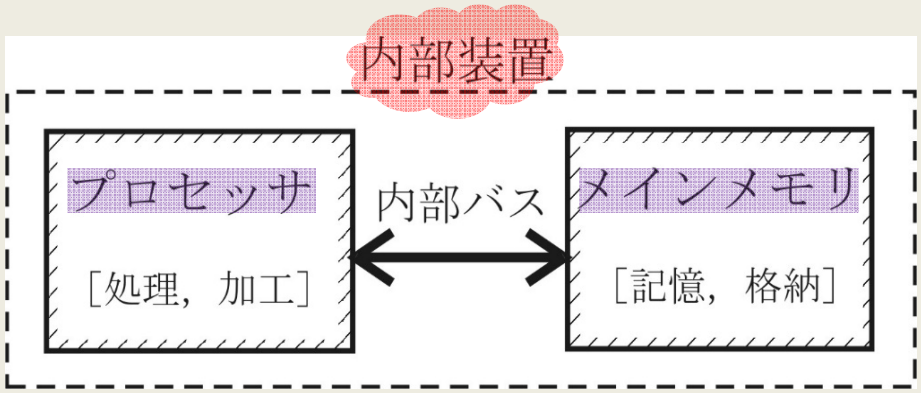
（重要）

プロセッサとメインメモリ

■ 内部装置（＝コンピュータ本体）
- ➢ 簡単な計算から高度な情報処理までを実行するハードウェア機構
- ➢ 次の2種類のハードウェア装置を組み合わせたハードウェアシステム

■ プロセッサ：情報を処理，その処理の仕方を制御

■ メインメモリ：情報を記憶，格納

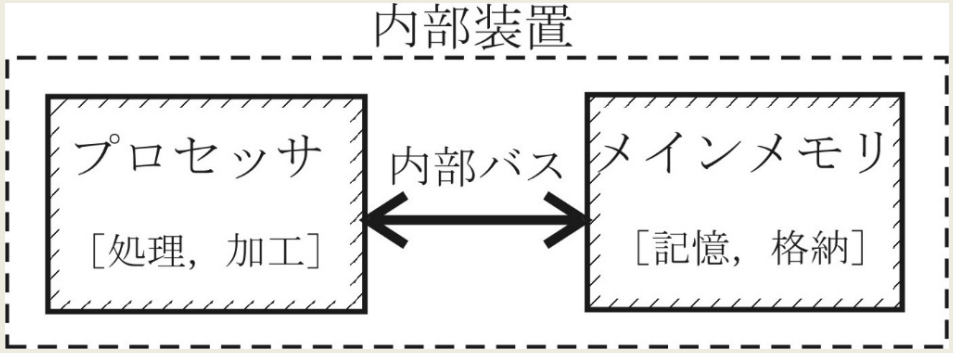


内部装置としての内部バス

- コンピュータが情報処理を行う際には，プロセッサとメインメモリとが情報のやりとり（＝通信）を行う必要

● 内部バス：プロセッサとメインメモリとの間を接続するバス (bus; 共用信号線)

＝プロセッサ－メモリバス，プロセッサバス，メモリバス



プロセッサ

- 情報の処理，その処理の仕方の制御という中心的な役割を果たす内部装置

  ＝CPU（Central Processing Unit; 中央処理装置）

  ＝内部装置全体（＝プロセッサとメインメモリとを併せて）

  ← 「コンピュータの中心的役割」を「情報の『記憶』能力を含むもの」と解釈

  - マイクロプロセッサ(microprocessor)：1個のIC上にほとんどのプロセッサ機能を搭載して構成したプロセッサIC

マシン語プログラム

- マシン語で直接書いたプログラム（命令とデータ）
  - プログラミング言語で書いたプログラム（＝プログラミング言語プログラム；例：Cプログラム）と区別するため
  - プログラミング言語プログラムや自然言語で入出力する命令やデータは，システムソフトウェアがあらかじめマシン語プログラムに変換

（再掲）

形（'0'/'1'の並び）は同じでも，意味が違う！

マシン語の実際

●コンピュータ（内部装置）では，情報（命令とデータ）はビット（列）か2進数かで表現

［意味］

➢ ビット（列）："0"か"1"の2種の記号（＝論理値）のいずれかを連結した記号（列）

← ［人間の世界］非数値
（例：文字/音声/画像/映像など）

◆ビット(bit)："0"か"1"の1個の記号，（参考）バイト(Byte) ＝ 8ビット

➢ 2進数：："0"か"1"の数字を1桁の重みを"2"とする規則に従って書き並べた数値

← ［人間の世界］10進数

➢マシン語 → ビット列か2進数（どちらも"0"/"1"の並び）で表現

内部装置における
プロセッサとメインメモリとの役割分担

■ マシン語プログラム（マシン命令とデータ）の使われ方

- プロセッサ：マシン命令にしたがって，データを処理（計算）
- メインメモリ：マシン語プログラム（マシン命令やデータ）を格納（記憶）

プロセッサによるデータ処理過程

- プロセッサは処理機能だけ，メインメモリは格納機能だけをそれぞれ装備

● プロセッサによる処理を指示するマシン命令も，処理される対象のデータそのものも，あらかじめメインメモリに格納しておく必要

◆ プロセッサがデータを処理するためには，その処理（計算）の仕方を指示するマシン命令と，その処理対象となるデータとを，メインメモリから取ってくる必要

◆ プロセッサによる処理が終われば，その処理結果をメインメモリに格納する必要

（重要）

プログラム内蔵

- （メイン）メモリというハードウェアにプログラムであるソフトウェア（マシン語プログラム：命令，データ）をあらかじめ格納
- プロセッサというハードウェアが必要時にそれらソフトウェアを呼び出して使用すること

■現代のコンピュータの構成上の原理：

- ハードウェアとソフトウェアの機能分担
- プロセッサと（メイン）メモリによる機能分担

→ 同時に実現

3 内部装置

3.1 コンピュータの内部装置

3.2 プロセッサとメインメモリとの通信

プロセッサーメインメモリ間通信(1)

[必要性]

(例)内部装置による加算(足し算)操作

1. 「加算」を指示するマシン命令をメインメモリからプロセッサに取ってくる
2. 加算の対象となるデータ(加算は2項演算であるから「加えられる数(被加数)」と「加える数(加数)」との2つが必要)をメインメモリからプロセッサに取ってくる
3. プロセッサが加算を行う
4. 加算の結果(和，これもデータ)をプロセッサからメインメモリへ格納する

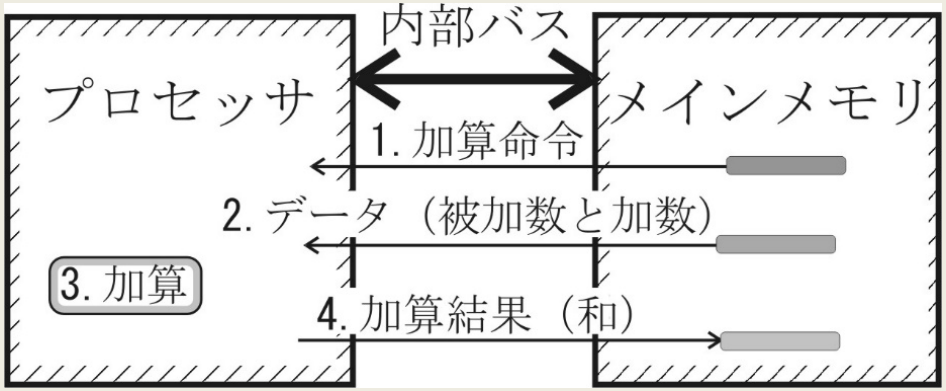
- 内部装置は処理(計算)をくり返す

プロセッサーメインメモリ間通信(2)

[実際]

- プロセッサとメインメモリとを結ぶ内部バス上を情報が転送
  - (1) メインメモリからプロセッサへ：マシン命令と2個の演算対象データ
  - (2) プロセッサからメインメモリへ：演算結果データ

➢ 内部バス：プロセッサとメインメモリ間の通信機構



マシン命令として示すべき情報(1)

- マシン命令はプロセッサに指令する言葉

(例)加算(足し算)操作

(1) 命令コード(code; 符号, 記号), 演算コード:「加算」という処理(計算)方法を指示する情報

- 命令コードで指定できる種類の演算機構をハードウェアとして装備する必要
- ICなどの限られたハードウェアを有効利用するために, 命令コードで指定できる演算種類は厳選

マシン命令として示すべき情報(2)

(2) 命令オペランド(operand)，演算オペランド，オペランド：「どのデータ（被加数）にどのデータ（加数）を足すのか？」という情報

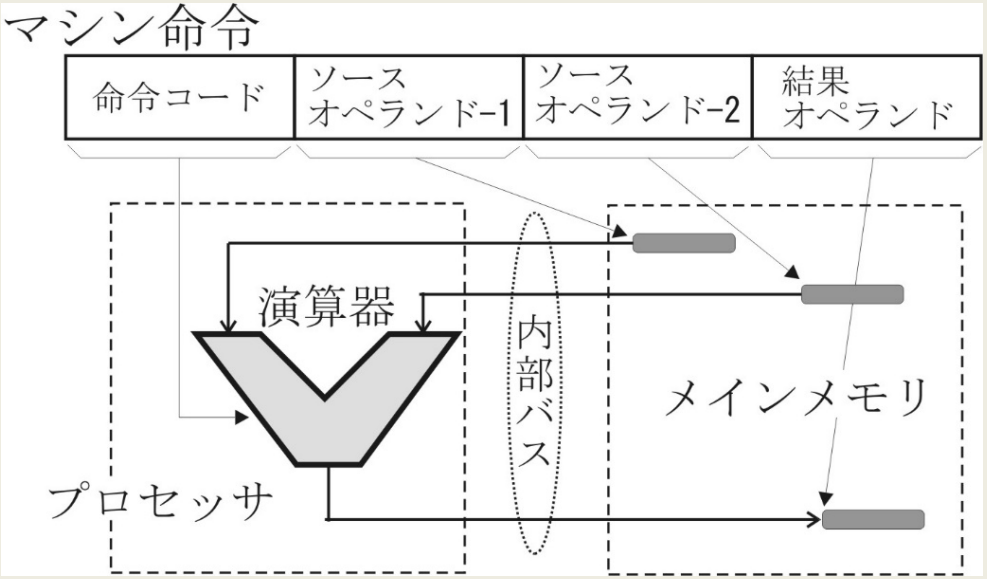
- ➢ 原則として，そのデータが存在する場所を指定
- ◆ソースオペランド(source operand)：演算（ここでは加算）されるデータの格納場所を指定するオペランド
- ◆結果オペランド：演算（ここでは加算）結果を格納する場所を指定するオペランド

マシン命令として示すべき情報(3)

- 2項演算を指示するマシン命令中に，合計4個の情報を埋め込む

(1) 1個の命令コード

(2) 2種類3個（2項演算の場合）の命令オペランド



データの指定方法(1)

■ データ：プロセッサで処理される情報

- 演算に使用されるデータやその格納場所はマシン命令中のオペランドとして指示
  - データの種類は，命令コードとは異なり，多種多様であり，少数に限定されず無数

データの指定方法(2)

- 加算命令（例）に使われるデータの種類：整数と実数，各々の数値は無数
- 2項演算（例）：1個のマシン命令中に命令オペランドは3個必要だが，3個ともデータとして直接マシン命令中に書くとマシン命令が長くなる

● 命令コードとオペランドの組み合わせでマシン命令を構成 → マシン命令の種類は数限りなく必要

（重要）

データの指定方法(3)

- オペランドは，原則として，データそのものとして示すのではなく，データの格納場所として指定

アドレス(address; 番地)

- オペランド：プロセッサが処理や計算を行う際のデータの格納場所をメインメモリのアドレス（＝メインメモリアドレス）として指示
- メインメモリアドレス（サイズは固定，1バイト(Byte; B)*単位が普通）は，欠番なし重複なしで順に付番

（* 1バイト＝8ビット）

メインメモリアドレス

- オペランドはメインメモリアドレスとして指定可
  - 種々のデータの組み合わせによる加算は，命令コードは「加算」にして，オペランドとして指定するメインメモリアドレスだけを変化させればよい
  - 「加算」（例）という命令コードを持つマシン命令は1種類だけ用意
- マシン命令もメインメモリに格納
  - マシン命令を取り出す際の指示も，オペランドと同様に，メインメモリアドレスで

コンピュータと人間との比較(3)
—メインメモリ(コンピュータ)と脳(人間)との相違—

■ メインメモリ(コンピュータの内部装置)
- 蓄積した情報(＝記憶)にアクセスするには，その場所(アドレス)を指定(指示)する必要 ← アドレスによるアクセス
- 人間が操作さえ誤らなければ，蓄積情報を失わない(＝忘却がない)

■ (人間の)脳
- 蓄積した情報(＝記憶)を，その一部の欠片から，瞬時に引き出せる(連想) ← 記憶内容によるアクセス
- 忘却によって情報や記憶を失う

命令実行サイクル(1)

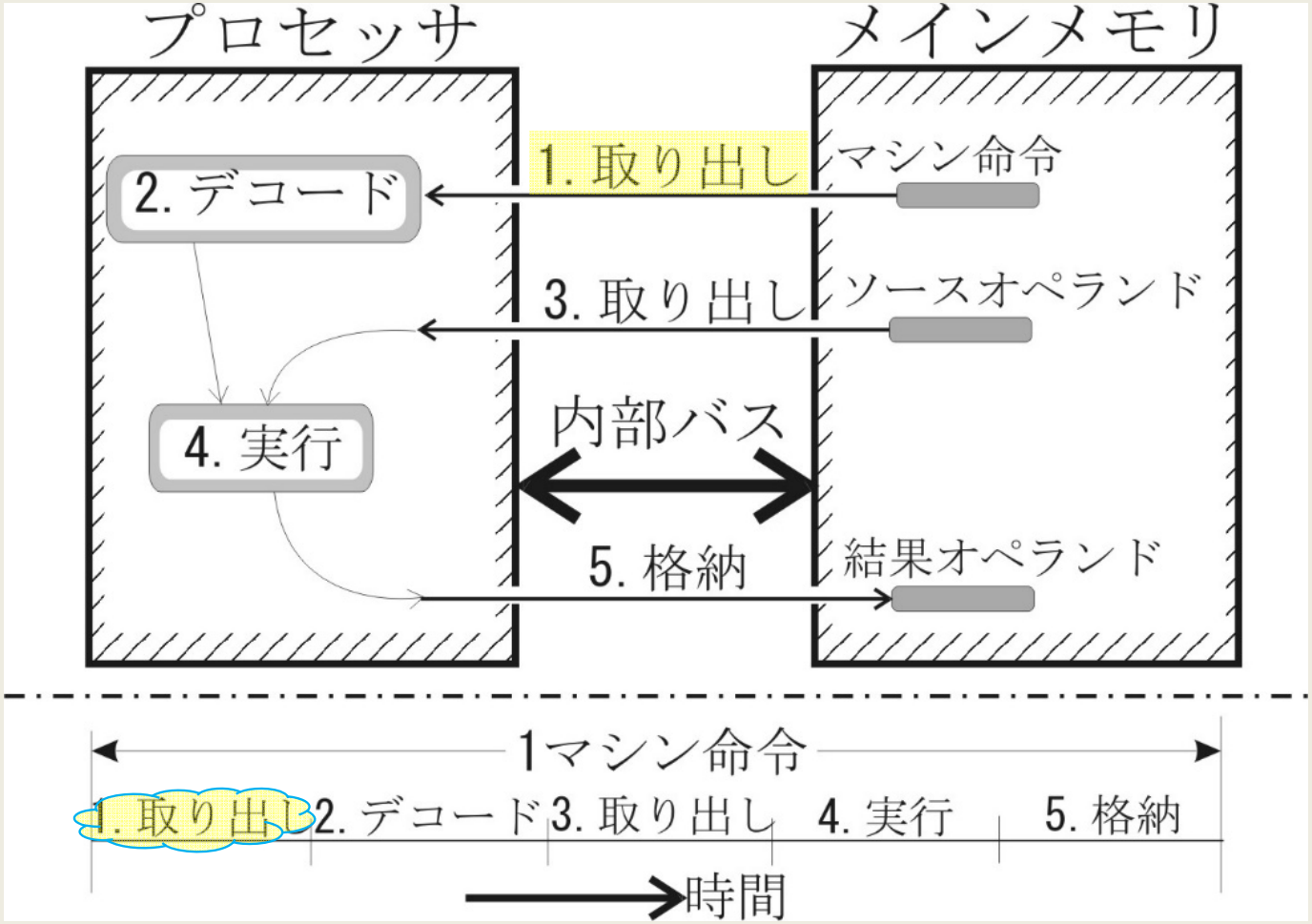
- ステージ(stage; 過程, 段階): マシン命令の各部分機能の実現過程
  - マシン命令の機能は共用する複数ステージによって実現
  - ステージごとに分担する機能は相異なる
  - ステージはこの順で実行する必要

命令実行サイクル(2)

■（例）加算（足し算）操作

1. 命令取り出し：メインメモリアドレスで指定されたメインメモリ中に格納されているマシン命令を1つ，内部バスを介してプロセッサ内に取り出す
   - ➢ 取り出されたマシン命令が今から実行すべき命令

命令実行サイクル（図）

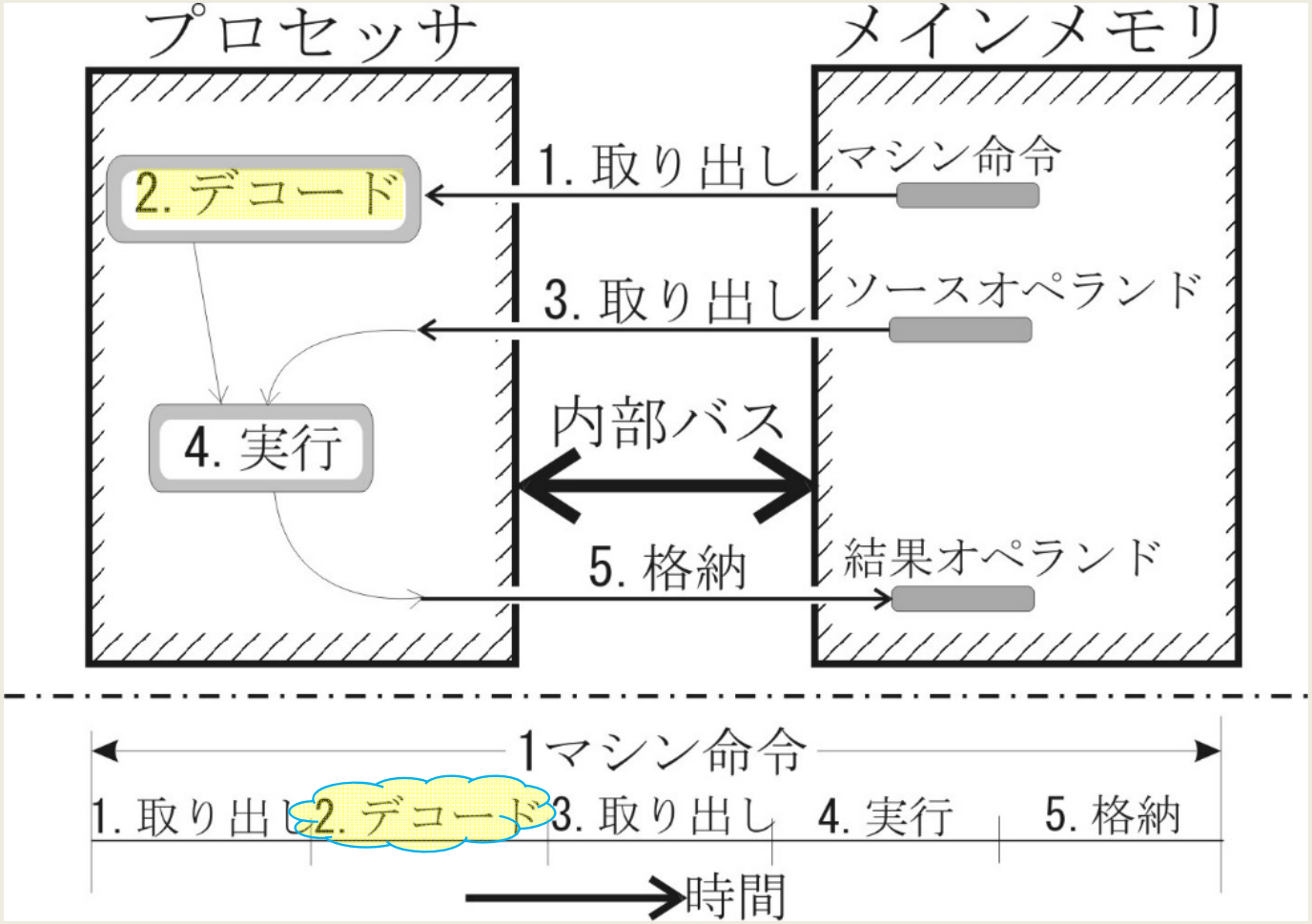

プロセッサ
メインメモリ
1. 取り出し
マシン命令
2. デコード
3. 取り出し
ソースオペランド
4. 実行
内部バス
5. 格納
結果オペランド
1マシン命令
1. 取り出し
2. デコード
3. 取り出し
4. 実行
5. 格納
時間

命令実行サイクル(3)

2. 命令デコード(decode; 復号化):1のステージで取り出されたマシン命令から命令コードやオペランドなどを分離
   - 「命令コードとオペランドとは，それぞれを使用するハードウェア機構が異なる」ために必要となる操作

     ◆(参考)命令を組み上げる操作であるエンコード(encode; 符号化)の逆操作

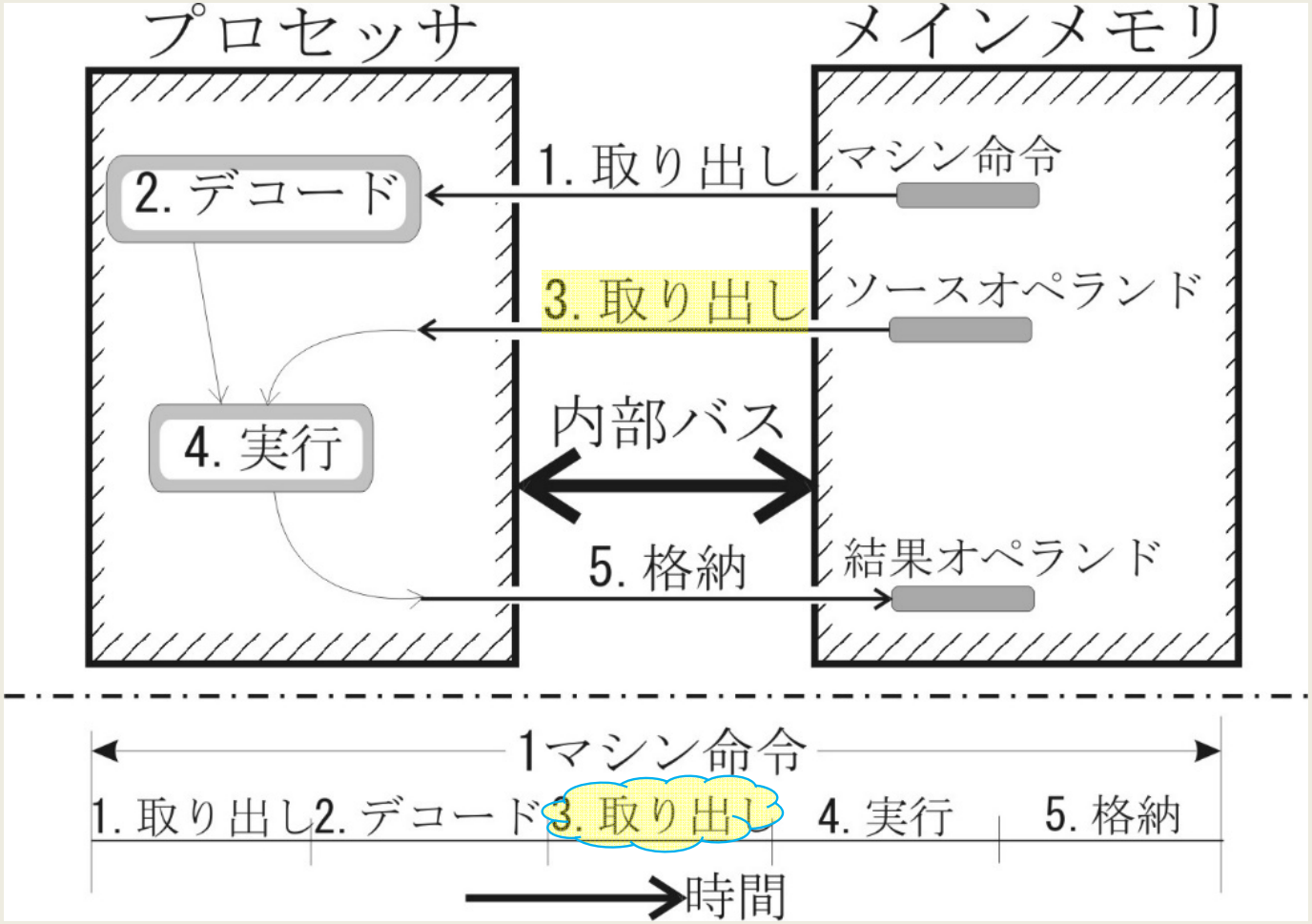
命令実行サイクル（図）


プロセッサ
メインメモリ
1. 取り出し
マシン命令
2. デコード
3. 取り出し
ソースオペランド
4. 実行
内部バス
5. 格納
結果オペランド
1マシン命令
1. 取り出し
2. デコード
3. 取り出し
4. 実行
5. 格納
時間

命令実行サイクル(4)

3. オペランド取り出し：2のステージで取り出されたソースオペランドによって指定されたメインメモリ中のデータを，内部バスを介してプロセッサ内に取り出す
   - その命令が必要とする個数（(例)2項演算なら2個）のデータを取り出してくる必要

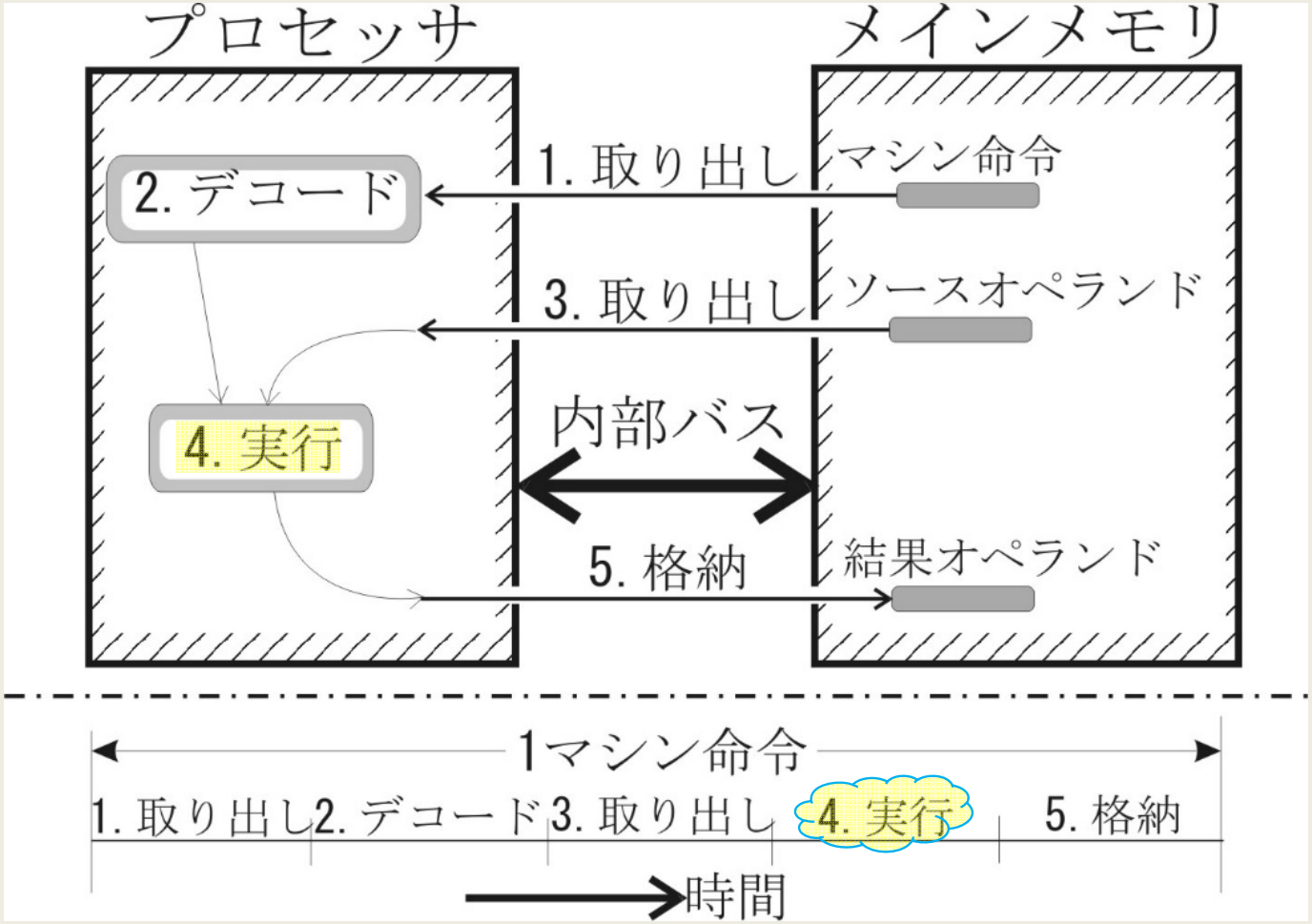
命令実行サイクル（図）


プロセッサ
メインメモリ
1. 取り出し
マシン命令
2. デコード
3. 取り出し
ソースオペランド
4. 実行
内部バス
5. 格納
結果オペランド
1マシン命令
1. 取り出し
2. デコード
3. 取り出し
4. 実行
5. 格納
時間

命令実行サイクル(5)

4. 実行：2のステージで取り出された命令コードにしたがって，3のステージで取り出されたデータに対して処理（計算）

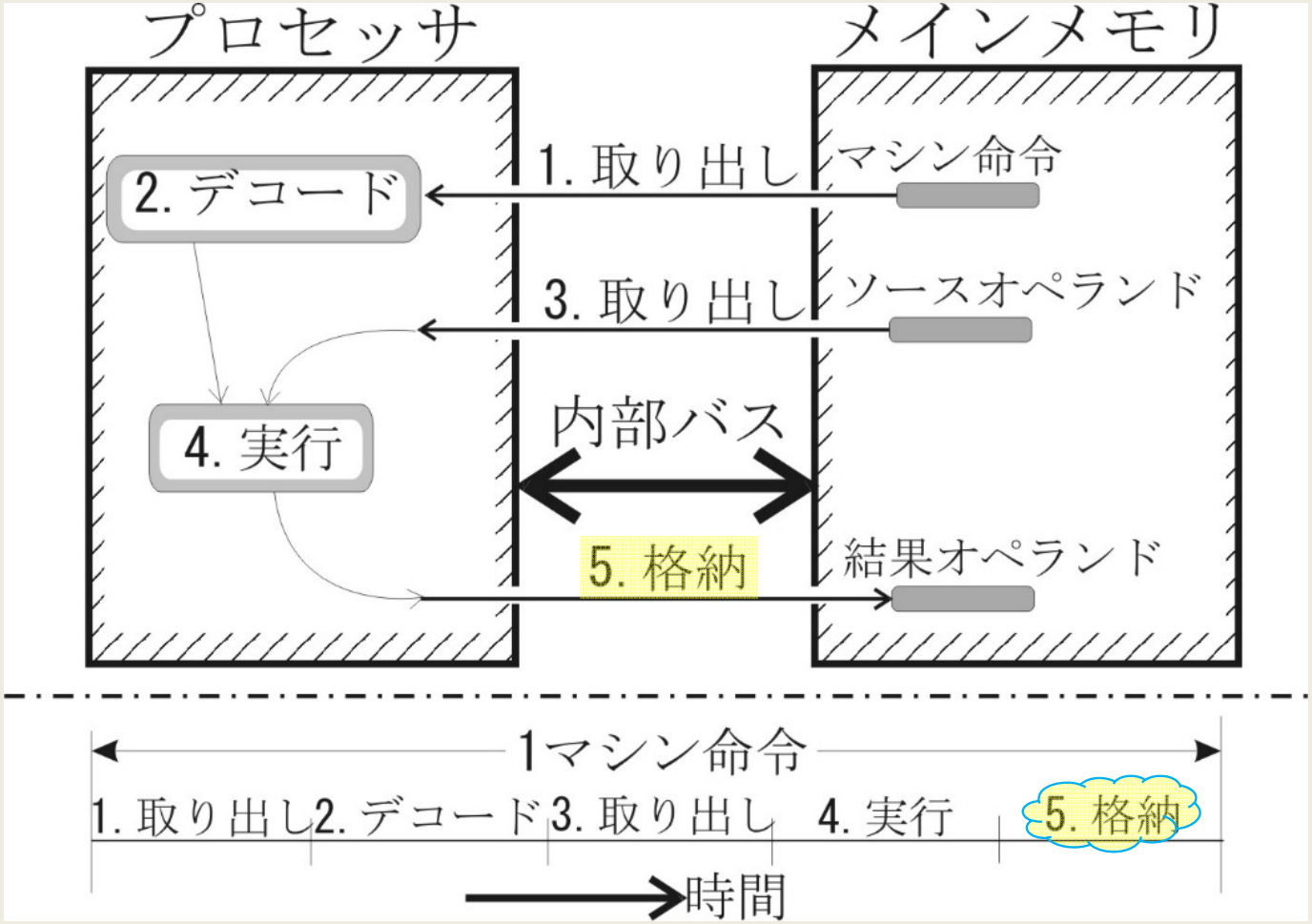
命令実行サイクル（図）


プロセッサ
メインメモリ
1. 取り出し
マシン命令
2. デコード
3. 取り出し
ソースオペランド
4. 実行
内部バス
5. 格納
結果オペランド
1マシン命令
1. 取り出し
2. デコード
3. 取り出し
4. 実行
5. 格納
時間

命令実行サイクル(6)

5. 結果格納：2のステージで取り出された結果オペランドによって指定されたメインメモリアドレスに，4のステージで生成された処理（計算）結果を内部バスを介して格納

命令実行サイクル（図）

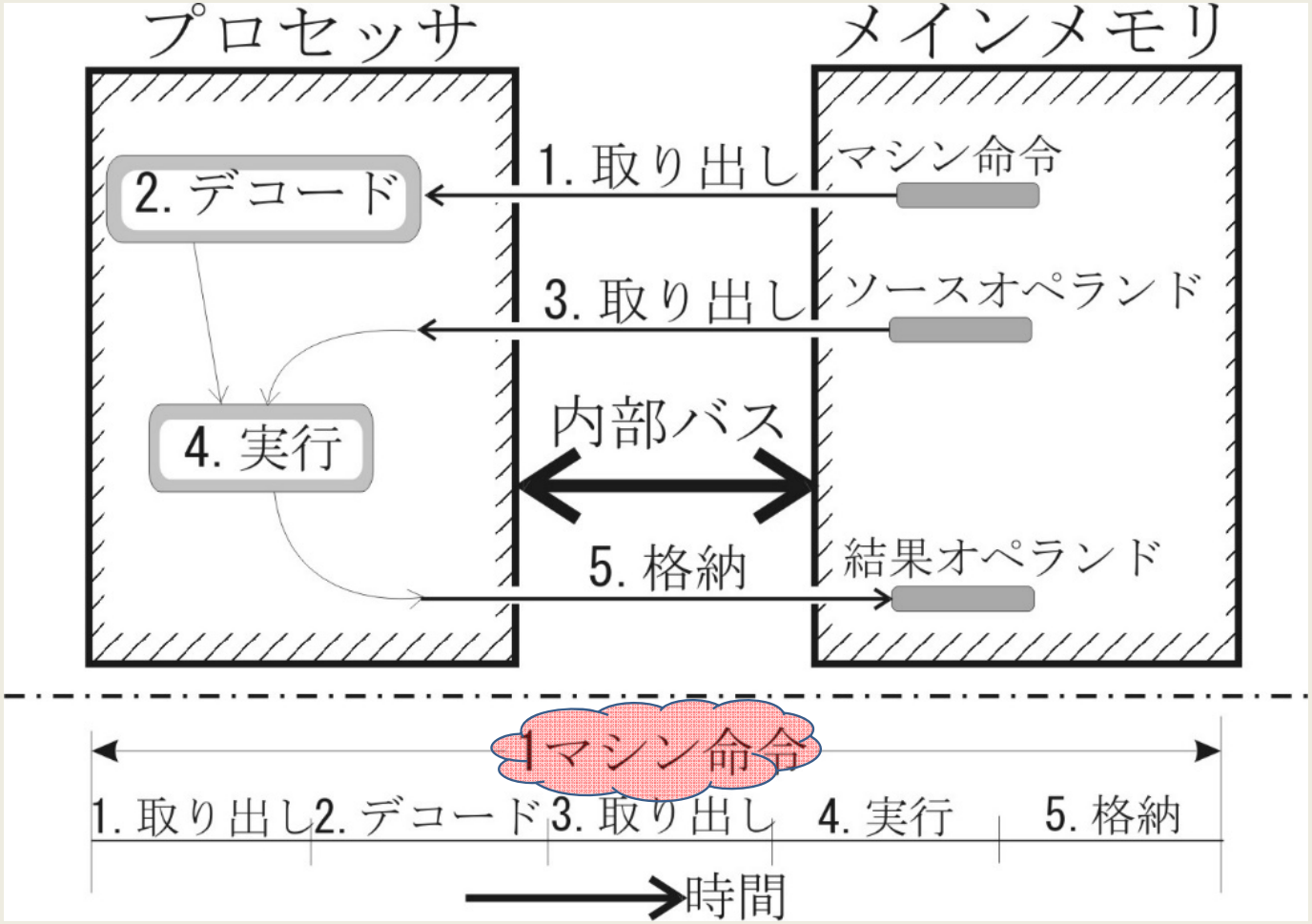

プロセッサ
メインメモリ
1. 取り出し
マシン命令
2. デコード
3. 取り出し
ソースオペランド
4. 実行
内部バス
5. 格納
結果オペランド
1マシン命令
1. 取り出し
2. デコード
3. 取り出し
4. 実行
5. 格納
時間

命令実行サイクル(7)

- 1～5の各ステージすべてをこの順で行って1マシン命令の実行が完了
- マシン命令列（マシン語プログラム）の実行は，この1～5のステージをマシン命令ごとにくり返す

■ 命令実行サイクル(cycle)：マシン命令機能（＝1～5のステージ）の実現ごとのくり返し

命令実行サイクル（図）


プロセッサ
メインメモリ
1. 取り出し
マシン命令
2. デコード
3. 取り出し
ソースオペランド
4. 実行
内部バス
5. 格納
結果オペランド
1マシン命令
1. 取り出し
2. デコード
3. 取り出し
4. 実行
5. 格納
時間