コンピュータのしくみ

吉川　雅弥
泉　　知論　共著

コロナ社

ま　え　が　き

我々の身の回りには，マイコンをはじめとしてさまざまなコンピュータがあり，日々の生活においてそれらのコンピュータを使っている。「どのようなものにコンピュータが使われているか？」という質問には，多くの人が即答できると思うが，「コンピュータはどのように動作するのか？」という質問には，答えに困る人が出てくると思う。そこで本書は，大学生や若手技術者を対象に，コンピュータのしくみを説明したものである。1 章ではコンピュータの構成や動作原理について概説し，2 章ではコンピュータの中での情報の表現方法について詳細に説明している。また，3 章ではコンピュータのさまざまな機能を実現する論理演算について解説し，4 章ではデータを保持する機構と，演算器や記憶装置の接続方法について説明している。また，5 章では演算について，その原理とハードウェアでの構成方法について説明し，6 章ではアセンブリ言語について説明している。このアセンブリ言語は，一般的に機種によって異なるが，本書では経済産業省の情報処理技術者試験の受験も考えて，当該試験で用いられる CASL II を使用している。最後に，7 章ではコンピュータの中の演算器などをどのように制御するかについて解説している。

すべての章を勉強することで，「コンピュータはこのように動作する」と，自信を持って答えられるようになると考える。大学や高専での教科書，若手技術者の参考書として利用していただければ幸甚である。本書では，著者らの大学におけるこれまでの講義での学生の理解度を鑑みて，動作原理などを解説している。重要な部分については，例題を付けているので，ぜひ解いてもらいたい。また，発展的な内容については「コーヒーブレイク」に書いてあるので，より掘り下げて学習したい場合には，「コーヒーブレイク」も熟読してもらいたい。

本書の執筆にあたっては，数多くの関連する専門書を参考にさせていただい

た。主要なものについては，関連する内容ごとに巻末に挙げている。漏れているものもあるかと思うが，その点はご容赦願いたい。

本書の出版にあたって，コロナ社の関係各位には叱咤激励も含めていろいろとお世話になり感謝したい。また，本書での図の作成にあたり，野崎佑典氏には全面的に協力していただいた。ここに深く感謝申し上げる。

2016 年 12 月

吉川雅弥，泉　知論

目　　次

1. コンピュータの基礎

2. 情報の表現

3. 論 理 の 世 界

4. 記 憶 と 接 続

5. 演　　算

6. コンピュータの言葉

7. 制 御

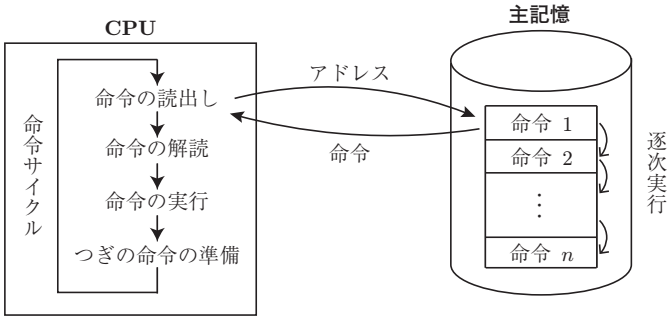

図 1.5　命令の実行方法

表 1.3　命令サイクル

処　理	ステップ	内　容
命令の取得	(1)	主記憶からの命令の読出し
	(2)	命令の解読
オペランドの処理	(3)	アドレスの計算
	(4)	オペランドの読出し
	(5)	命令の実行
	(6)	結果の格納
つぎの命令の準備	(7)	プログラムカウンタの更新

命令をデコーダを使用して解読する。デコーダとは，あらかじめ用意された命令に対して，制御信号を選択する回路である。この選択された信号によって制御装置が当該命令に対応する処理を実行する。

つぎに，(3)〜(6) のオペランドの処理では，オペランドの読出しや結果の書込みに必要なアドレスを計算する。計算したアドレスを利用して，主記憶またはレジスタからオペランドを読み出す。その後，解読した命令の内容（例えば，加減算，乗除算などの四則演算および論理演算）に従い，データに対する演算や加工を行い，その結果を主記憶またはレジスタに格納する。最後に，つぎの命令のアドレスを計算し，プログラムカウンタに設定する。

基本的な動作は，(1)〜(7) の繰返しで，実際のコンピュータでは，いくつかのステップを同時に実行する。これらの処理ステップの制御を行う制御装置には，二つの構成方式がある。一つはマイクロプログラム制御方式（microprogrammed control）で，もう一つは，布線論理制御方式（wired logic control）である。

さい。

(3) $S=\sum_{n=1}^{\infty}\frac{1}{n!}$ で生じる誤差は，どのような誤差か答えなさい。

【解答】

(1) 有効桁数が失われる桁落ちの誤差が生じる可能性がある。

(2) $x \gg y$ では，$x+y=x$ となる。すなわち，y は，x と同じ扱いになり，無視される。したがって，このような誤差は情報落ちである。

(3) $1/n!$ が循環小数になる場合があり，丸め誤差が生じる。さらに，無限に $1/n!$ を加算することはできないため，有限回で処理を打ち切り，打切り誤差が生じる。 ♢

章　末　問　題

【1】 10 進数 672 を，2 進数と 16 進数に変換しなさい。

【2】 16 進数 E4A を，8 進数に変換しなさい。

【3】 10110010 が「1 の補数表現」の場合，いくつの値を示しているか 10 進数で答えなさい。また，2 の補数表現の場合についても答えなさい。

【4】 2 の補数表現の場合，1 の補数表現より表現できる数が多い。その理由を簡潔に説明しなさい。

【5】 1 の補数表現において，循環桁上りの処理をしないと，正しい演算結果が得られない理由について，簡潔に述べなさい。

【6】 10 進数 0.7 を，2 進数に変換しなさい。

【7】 9 進数 12.34 を，3 進数に変換しなさい。

【8】 つぎのビットパターンが IEEE 754 の単精度で表現されている場合，いくつの値を示しているか 10 進数で答えなさい。

(1) 01010010110110000000000000000000

(2) 11000111101110100000000000000000

【9】 10 進数 6.125 と 10 進数 5.75 を，2 進数の浮動小数点方式に変換して，加算を行いなさい。ただし，仮数の桁数を 4 桁として，丸めの方法は，零捨一入とする。

【10】 10 進数 −2.25 と 10 進数 0.4375 を，2 進数の浮動小数点方式に変換して，乗算を行いなさい。ただし，仮数の桁数を 4 桁として，丸めの方法は，零捨一入

の多数決を取る関数となり，それぞれ論理式は式 (5.1)，式 (5.2) になる。論理回路を**図 5.4** に示す。

$$s_i = a_i \oplus b_i \oplus c_i = a_i\overline{b_i}\,\overline{c_i} + \overline{a_i}b_i\overline{c_i} + \overline{a_i}\,\overline{b_i}c_i + a_ib_ic_i \tag{5.1}$$

$$c_{i+1} = a_ib_i + b_ic_i + c_ia_i \tag{5.2}$$

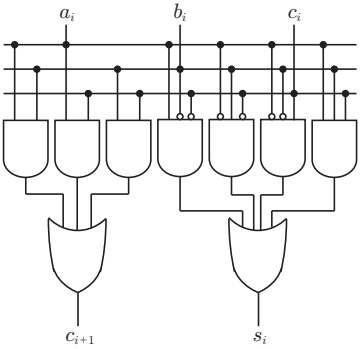

図 5.4　1 ビット全加算器の論理回路

1 ビットの全加算器を並べ，桁上げを下位から上位に 1 桁ずつつなぐことで多ビットの加算器が実現できる。**図 5.5** に 4 ビット加算器の構成を例示する。このような構成の加算器を桁上げ伝搬加算器（ripple carry adder，RCA，ripple は「さざ波」の意）と呼ぶ。

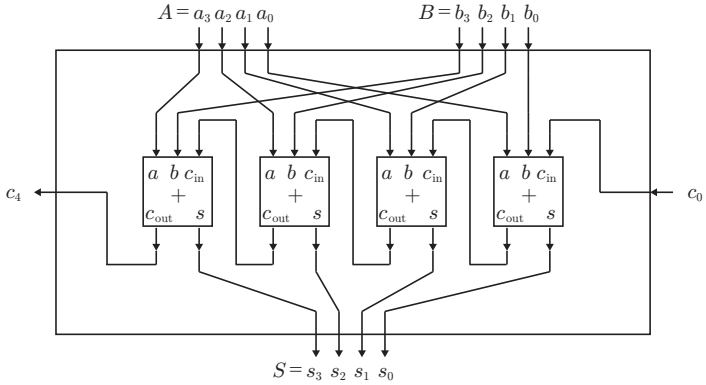

図 5.5　4 ビットの桁上げ伝搬加算器

```
1：　　LD　　GR1，NUM　　；ラベル NUM のアドレスの内容を GR1 へ
2：　　ADDA　　GR1，GR2　　；GR1 と GR2 を加算
```

図 6.28　機械語変換用の例題

つぎに，2 行目の ADDA 命令は表 6.8 より 1 語構成の命令となる。15･･･8（8 ビット）は，00100100 で，7･･･4（4 ビット）と 3･･･0（4 ビット）は，それぞれ GR1 と GR2 に対応するため，0001 と 0010 となる。16 進数表記をすれば 2412 となる。このようにアセンブリ言語から機械語へ変換することをアセンブルするという。

インデックスレジスタに GR0 が使用できない理由は，使用しない場合を 0 としているためで，もし GR0 をインデックスレジスタとして使用した場合，未使用の 0 か GR0 かの違いを区別することができない。

例題 6.3　機械語への変換について，つぎの問いに答えなさい。

(1) **図 6.29** はあるプログラムの一部である。表 6.8 を用いて，図 6.29 を機械語に変換しなさい。ただし，ラベル N1，N2，N3 の示すアドレスをそれぞれ 000Ah，000Bh，000Ch とする。

```
1：　　LD　　GR1，N1
2：　　LD　　GR2，N2
3：　　ADDA　　GR1，GR2
4：　　ST　　GR1，N3
5：　　RET
```

図 6.29　アセンブリ言語から機械語への練習問題

(2) つぎの機械語を CASL II のアセンブリ言語に変換しなさい。ただし，各命令は，16 進数で表記されているものとする。

(a) 3456

(b) 52200008

(c) 10730004

することができない。

(3) パイプライン処理による性能向上を実現するためには，十分な数の命令が必要である。

(4) パイプライン処理によって，レイテンシを改善することができる。

【解答】

(1) 間違い。理想的な状態で n 倍になる。そのため，現実的には数が多ければ n 倍に近づく。

(2) 間違い。すべてのステージの処理時間が同じほうが，性能が向上しやすいのであって，同じでなければパイプライン化できないわけではない。

(3) 正しい。

(4) 間違い。パイプライン処理で改善するのは，スループット。

したがって，答えは，(3)。　　◇

パイプライン処理は，図 7.5 に示したように，ある命令を実行しているつぎのクロックサイクルで，つぎの命令を実行する。しかし，これを実行できない場合がある。これをハザードと呼び，構造ハザード，データハザード，制御ハザードの三つがある。

まず，構造ハザードは，並列に実行されている命令に，演算器，レジスタ，メモリなどのハードウェアのリソースが不足している場合に起きる。**図 7.6** に構造ハザードの例を示す。図の例では，メモリへのデータの書込み（または読出し）するためのメモリへのアクセス（MEM）と，命令をフェッチするためのメモリへのアクセス（IF）で競合が起きている。

つぎに，データハザードは，先行命令の演算結果を後続の命令が利用する場

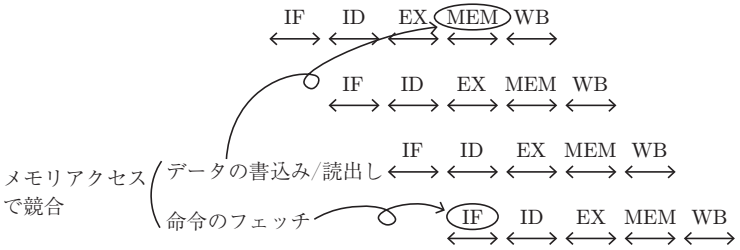

図 7.6　メモリへのアクセスでの構造ハザード

索　　引

―― 著 者 略 歴 ――

吉川　雅弥（よしかわ　まさや）
2001 年　立命館大学大学院理工学研究科博士後期課程修了，博士（工学）
1998 年　立命館大学第一号助手
2001 年　立命館大学博士研究員
2004 年　立命館大学講師
2007 年　名城大学准教授
2012 年　名城大学教授
現在に至る

泉　　知論（いずみ　とものり）
1998 年　東京工業大学大学院理工学研究科博士後期課程修了，博士（工学）
1998 年　京都大学助手
2005 年　立命館大学助教授
2007 年　立命館大学准教授
2016 年　立命館大学教授
現在に至る

コンピュータのしくみ

Mechanism of Computers

2017 年 2 月 27 日　初版第 1 刷発行　★

検印省略

著　者　吉川　雅弥
　　泉　　知論
発行者　株式会社　コロナ社
　　代表者　牛来真也
印刷所　三美印刷株式会社

112–0011　東京都文京区千石 4–46–10
発行所　株式会社　コロナ社
CORONA PUBLISHING CO., LTD.
Tokyo Japan
振替 00140–8–14844・電話(03)3941–3131(代)
ホームページ　http://www.coronasha.co.jp

ISBN 978–4–339–02867–6　（新井）　（製本：愛千製本所）
Printed in Japan