コンピューターとプログラム

山本昌志*

2005 年 10 月 14 日

1　本年度，学習することと目標

本年度，3 年生の電子計算機では，アセンブラ言語を学習する．それにはにはいろいろあるが，ここでは情報処理技術者試験で使われる「CASL II」という言語を学習する．

1.1　2 年生で学習したこと

前年の授業，2 年生の「電子計算機」とのかかわりについて，簡単に述べておく．2 年生と 3 年生の「電子計算機」を通して，コンピューターの基本的な仕組みを理解することがこれらの講義の目的である．2 年生の「電子計算機」の授業で学習したことは，大体

- 記数法
- ブール代数
- 論理回路

のようなことであったはずである．ここで，学習したことで最も重要なことは，

- いかなる論理演算 (真理値表で表せる) であろうとも，論理和 (OR) と論理積 (AND)，否定 (NOT) のゲートで作ることが可能である

ということである．これらの例として，最後に加算の回路を学習したであろう．皆さんは，加算にかかわらずどんな論理の回路でも設計できまるはずである．言い方を変えると，どんな演算回路でも皆さんは設計できるということである．それも，たった 3 種類の素子 (OR, AND, NOT) だけで，可能なのであるから驚きである．

1.2　3 年生で学習すること

1.2.1　講義内容

この講義の目的は，

*国立秋田工業高等専門学校　電気工学科

- コンピューターの基本的な仕組みを理解する
- 基本情報処理技術者試験に合格する

である．これらの目的を達成するために，諸君には多くの課題を課すことにする．与えられた課題を地道にこなし，コンピューターと言うものを理解して欲しい．

これら，二つの目的のうち，後者は分かる．CASL II が基本情報処理技術者試験の科目となっているからである．1 年生の時に学習した FROTRAN は，科目となっていない．C 言語を知らない者が，これに合格するには，CASL II を勉強するのがもっとも近道である．4 月と 10 月の 2 回/年，試験が実施されるので，気概のある者はトライせよ．ただし，これに合格するためには，この講義だけでは不足でもう少し勉強が必要である．

FORTRAN を学習しても，コンピューターの仕組みは分からない．C 言語でもほとんど分からないだろう．一方，アセンブラ言語の場合，コンピューターの仕組みが分からないと，この言語でのプログラムは不可能である．なぜならば，アセンブラ言語はコンピューター (CPU) の動作を記述する言語であるからである．まだ，諸君にはこのことの意味は分からないだろうが，本日の授業の後半で，このことを述べる．

先に示した目標に達成するために，以下のような内容の講義を進める．

1. コンピューターの基本的な構造を学ぶ
2. コンピューター内部での文字や数字の表現方法を学ぶ
3. アセンブラ言語の命令の種類と基本動作を学ぶ
4. アセンブラ言語のプログラムの方法を学ぶ

1.2.2 この講義を受けて得られるもの

本心を言うと，次のようなことが理解できれば，この講義を受けた価値はあると思う．

- コンピューター (電子計算機) の仕組みを理解する．
- 0 と 1 で書かれた機械語とコンピューターの頭脳といわれる Central Processing Unit(CPU) の関係を理解する．
- 0 と 1 で書かれた機械語と C 言語や FORTRAN のような高級言語との関係を理解する．

そして，3 月始めの学年末試験で合格した者が実際に得られるものは，

- 単位
- コンピューターの仕組みは，簡単だなーという実感
- アセンブラが理解できるという優越感

くらいでしょうか．落第したものは，この反対のものが得られるであろう．

2　コンピューターには何が必要か

2.1　記憶する回路

本日の授業の後半で示すが，コンピューターは演算 (CPU) と記憶 (メモリー) の回路から成り立っている．記憶の回路については，4 年生で学習することになっている．ただ，簡単に書くと次のような図 1 のような入出力線があり，次に示す動作をすると思えばよい．

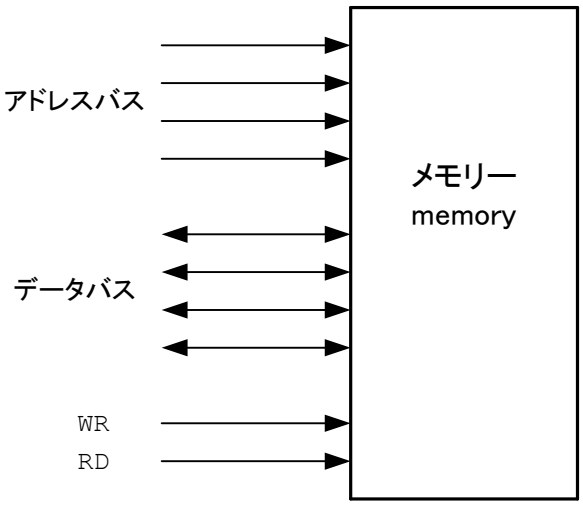

図 1: メモリーのモデル (ハードウェアー)

アドレス	データ
0000	0000
0001	0101
0010	1000
0011	0100
0100	0001
0101	1001
0110	0011
0111	0110
1000	1111
1001	0000
1010	1011
1011	0001
1100	1100
1101	0000
1110	0010
1111	0111

図 2: メモリーのモデル (概念)

この図のアドレスバスとデータバス，WR，RD と書かれた線は，電圧を印加するリード線である．それぞれは，以下の役割があり，データを記憶する．

データバス データを読み書きする場所を示すための線．この場所のことをアドレスと言う．ここでは 4 本の線が有るので，記憶場所は 16 個ある．

アドレスバス 読み書きのときのデータを受け渡すための線．4 本有るので 4 ビットのデータの受け渡しが可能である．

WR メモリーにデータを書くと指令するための線．

RD メモリーからデータを読み出すと指令する線．

この回路の仕組みは皆さんは理解できないと思うが，動作は理解しなくてはならない．動作は簡単なので，このような回路を作ると巨万の富が得られると思うと挑戦したくなるだろう．今からでは遅いが，半世

紀ほど前であれば大富豪も夢ではない．どうやって作るかよりも，なにを作るかの方が重要ということが分かるであろう．世界の先端に居る人は，なにを作るべきかということが分かり，そして巨万の富を得るのである．

2.2 コンピューターに必要なもの

これで，コンピューターを作るうえでの基本ハードウェアー，演算とメモリーの回路が揃った．演算の回路はここで示していないが，諸君が 2 年生の時に学習した組み合わせ回路から構成されるものと考えてよい．しかし，これだけでは計算はできない．この状態は，人間で言うと赤ちゃんみたいなものである．脳はあるが未だなにもできない．不足しているものは，

- 演算といっても，必要な演算が分からない．計算に必要な演算回路を決める必要がある．人間で言うと，教育を行い，脳の神経細胞をつなぐことに相当する．
- 実際の計算を行わせるためには、プログラムが必要である．他の人に計算をさせるためには指示が必要なのと同じ．

です．必要な演算回路とプログラムを作らないと計算ができない．この講義では，演算回路とプログラムについて，学習することになる．

この 3 年生の「電子計算機」の講義を受けると，コンピューターの設計に必要な基礎的なことの全てを学んだことになる．今，巷で評判になっている「CPU の創り方」[1]という本の内容が理解できるはずである．

3 コンピューターのモデル

この辺の話は,「Interface 2002 年 9 月号」と「ファインマン　計算機科学」,「数学セミナー　2003 年 12 月号」を参考にしている．

3.1 コンピューターの動作

コンピューターの基本的な動作は，非常に簡単である．1 人の学生に登場してもらって，CPU の役割を担ってもらう．私が用意した小さな箱が，メモリーである．それに格納されている命令に従って，簡単な計算

- まずは，2×3 の計算

を行い，コンピューターの動作の基本を理解してもらう．

この計算を行うこのプログラムは，次のとおりである．

00 番地 08 番地のデータを CPU の記憶領域 a に格納する．

01 番地 CPU の記憶領域 b の値を 1 にする．

[1]渡波郁著　朝日コミュニケーションズ発行　ISBN4-8399-0986-5

02 番地 記憶領域 b から 09 番地の値を引き算する．結果が 0 ならば，06 番地の命令を実行する．

03 番地 記憶領域 a の値と 08 番地の値を加算して，記憶領域 a に格納する．

04 番地 記憶領域 b の値を 1 増加させる．

05 番地 02 番地の命令を実行する．

06 番地 記憶領域 a の値を 10 番地に格納する．

07 番地 おわり．

08 番地 2 が格納されている．

09 番地 3 が格納されている．

10 番地 計算につかう記憶領域．

気が付いたと思うが，足し算を繰り返すことにより，掛け算をの演算を実行しているのである．普通の CPU は乗算ができないのである．

実際ののコンピューターの動作は，これをとてつもない速度で実行している．パソコンの 3GHz の CPU では，数クロックで 1 つの命令を実行するので，1 秒間に億のオーダーの命令を実行できる．またメモリーは，512Mbyte で，約 5 億個の記憶領域が用意されている．コンピューターは単純な動作しかできないが，非常に高速で，大量のデータを処理する．これにより複雑なことを行っているようにみえるだけである．

このコンピューターのモデルで理解して欲しいことは，

- プログラムのデータもメモリ上に格納される．
- 命令に従い，データを書き換えている．

である．

3.2 チューリング機械

先に生徒諸君にコンピューターの動作を行ってもらったように，コンピューターというものは「メモリ中のデータとコンピューターの内部状態に従い，メモリーのデータを逐次的に書き換える計算機」と定義できる．こういうものを世界で最初に提案したのは，当時，ケンブリッジ大学の大学院生であったアラン・チューリング[2]ということらしい．1930 年代の半ばのことである．

現在，コンピューターの理論的モデルと言われるのが，図 3 に示すチューリング機械である．その特徴は，次の通りである．

- 書き換え可能な無限に長いテープと，オートマトンと言われる移動可能な機械からできている．
- テープには，いろいろな記号が書かれている．

[2]第二次大戦中はドイツの暗号装置エニグマの解析をしていた．

- オートマトンには，テープの内容を読み書き可能なヘッドと内部状態を記憶する装置，テープの任意の位置に移動する装置から構成されている．
- オートマトンの動作 (テープの読み書き) や移動は，今の場所のテープの記号と内部状態により決まる．

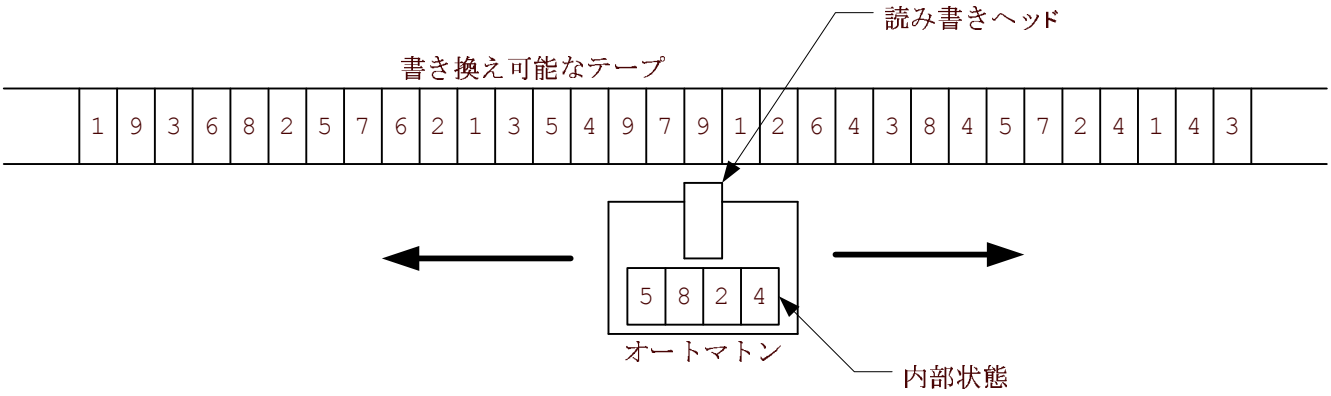

図 3: チューリング機械

これが，コンピューターのモデルである．今は，分からなくても，そのうち理解できるであろう．書き換え可能なテープはメモリーに，オートマトンは CPU に相当する．テープに書かれた記号は，プログラムであったりデータであったりする．内部状態はレジスタ[3]の値に対応する．

あるときチューリング機械が，図 3 の状態であったとする．テープの内容を読むあるいは書き直す，内部状態を変える，移動することのどれかが次の動作になる．次の動作は，テープの内容と内部状態により一意に決まる．要するに，テープの上を行ったり来たりして，内部状態を変えたり，テープの内容を読み書きしている自動機械がチューリング機械である．

このような動作をするチューリング機械で，どんなことができるのか?．このような単純な機械で，ありとあらゆる計算ができるのである．諸君が今まで学習してきた計算は，記号の操作の繰り返しになっている．人間の脳で計算するときも，計算と言えば記号の操作の繰り返しのはずである．要するに，チューリング機械ではこの種の計算が，可能なわけである．ただ，チューリング機械で計算できないものもある．この問題は，込み入って複雑なので，ここでは取り扱わないことにする．

以上で，チューリング機械の概要が分かったと思う．要するに言いたいことは，計算するという動作は，チューリング機械で表現できると言うことである．計算するという一見，知的な作業が，おもちゃのようなチューリング機械で表せるとは驚くべきことである．

3.3 ノイマン型コンピューター

ここで，少しコンピューターの発明されたころの話をしておこう．1940 年頃，ベル研にコンプレックスカリキュレータというものがあり，それはプログラムは人間が手で入れるもので，電卓と同じようなもので

[3]CPU のメモリーと思ってください

ある．つまり，プログラムは，それを操作する人の頭にあり，人の手がインターフェースとなっている．これでは，とても高速にプログラムの実行ができない．

次の進歩は，プログラムが自動で計算機に送り込まれ，それに従い実行される機械の発明である．ドイツの Z3 とハーバード大学のマーク I がこれにあたる．これらの機械は，計算はリレーで行われていた．プログラムは，Z3 の場合はフィルムに，マーク I の場合は紙テープに書かれていた．この場合、プログラムの実行速度は，フィルムや紙テープの読み取り速度で制限される．これは，高速に計算する上で非常に大きな問題であった．

1946 年，砲弾の弾道計算用に ENIAC と名づけられた真空管式の電子計算機が開発された．当時としては，とてつもない速度で計算することができる機械であったが，大きな問題があった．計算の内容，現在でいうプログラムを変えるとき，それはプログラムボード呼ばれる配線板上の配線を組み替える必要がった．この作業は大変で，1 日程度の時間が必要であったようである．当然，次のコンピューターを作るとき，この点の改良が議論されたのは言うまでも無い．プログラムの変更が大変ではあるが，先の紙テープのように外部からプログラムを送るのではなく，本体に内蔵していた点では大きな進歩である．これにより，高速に計算ができた．

次の EDVAC というコンピューターの開発では，プログラムを配線ではなく，メモリーの中に入れることが議論された．こうすることにより，プログラムの変更が容易でかつ高速で計算するコンピューターが可能となる．この開発の中に，天才数学者ノイマンがおり，以降，このようにプログラムを内蔵したものをノイマン式コンピューターと言う．ただし，このアイディアを出したのがノイマンかと言われると，定かではないようである．このコンピューターを実現するためのメモリーの開発は大変だった．

紆余曲折の後，プログラムと計算処理の対象であるデータは，同じ メモリー上に置かれるようになった．このように，同じ メモリー上に命令とデータがあるようなものをノイマン型コンピューターと言う．世界中のほとんどのコンピューターがこのノイマン型のコンピューターで，

- 1 次元的に並んだメモリーがあり，そこにプログラム (命令) もデータも格納される．メモリーの内容は，自然数の番地で参照できる．
- メモリーに格納されたプログラム (命令) とデータの見かけ上の区別はない．プログラムをデータとして見ることも，データをプログラムとしてみることもできる．

の特徴をもっている．チューリング機械そのものです．

プログラムとデータを別のメモリーに置く、コンピューター (CPU) もある．このような方式をハーバードアーキテクチャーと言う．

4 コンピュータープログラムの歴史

4.1 なぜプログラムが必要か?

所詮，コンピューターはプログラムの通りに動作する自動計算機に過ぎない．もし，プログラムが無かったらどうなるだろうか?．コンピューターは，全く動作せず，何もできないことは容易に想像がつく．いかにすばらしいハードウェアーがあっても，プログラムが無いと，それはただの箱である．

プログラムの無い計算機といえば，電卓を使うことを思い浮かべるであろう．電卓のみを用いて，例えば連立方程式を計算する場合を想像してみよう．この場合，人間の頭にその解法があり，それに従い手を動

かして計算をすることになる．この場合，プログラムに相当しているものが，頭の中にあり，指を通して，ハードウェアーである計算機に指示を与えてることになる．

では，いったいプログラムとは何だろうか?．一言で言うと，それはコンピューターに指示を与えるものである．その特徴は，

- あいまいな表現が許されない．

ことにある．機械相手ですから，仕方ないことである．

4.2 プログラム方法の変遷

未完に終わったが，最初にコンピューターを考えたと言われるのが，イギリスのチャールズ・バベッジ (Charles Babbage, 1791-1871) である．当時，イギリスは産業革命で，大量の織物が作られていた．それには，ジャカールが発明した自動織機が大活躍していた．織物の柄はパンチカードに書かれており，それに従い機械が自動的に織物を作っていた．バベッジはそれを利用して，自動計算機を作ろうと考えた．ただ，その機械を実現するための技術が当時では未発達で、バベッジの考えた自動計算機は実現できなかった．

最初に稼動した電子計算機は，アメリカの ENIAC である．それは，プログラムボード呼ばれる配線板上の配線を組み替える方法で，プログラムが書かれていた．これは，ハードウェアーそのものを変えることにより，プログラムしていることになる．要するにコンピューターの動作を配線で示したわけである．

もう少し進化すると，メモリーにプログラムを書くようになった．0 と 1 を全てメモリー上にスイッチで指定するようなことが行われていたようである．コンピューターの動作を 0 と 1 で示したのである．これこそが機械語である．今もそうであるが，コンピューターは 0 と 1 でできた機械語しか理解できない．当時は、それをそのまま指定していた．例えば，足し算をするときは，

```
0010 0000 0001 0000 0000 0000 0000 1010
```

と書く．これは，メモリーの内容そのもので，これに従い CPU の線の電圧が 0 や 1 になる．それに従い論理回路が動作し、計算をおこなうのである．

さすがに 0 と 1 を並べただけのプログラムは，分かりにくく，専門的なプログラマーにしかプログラムができなかった．これをもう少し，分かりやすくしたのがアセンブラ言語である．先ほどの加算のプログラムを

```
ADDA    GR1,ADDRESS
```

と書く．ADD というのは加算するという英語である．その後の A は気にしないこと．これは，GR1 と ADDRESS を加算せよという命令となっている．詳細は後ほど学習するが，先ほどの 0 と 1 が並んだプログラムよりは格段に分かりやすい．これは，

- ADDA GR1　→　0010 0000 0001 0000
- ADDRESS　→　0000 0000 0000 1010

に対応している．要するに，人間がわかりやすいアセンブラ言語は，コンピューターが唯一理解できる機械語と 1 対 1 に対応しているのである．このようにすることで，プログラムは格段に容易になる．

ただし，アセンブラ言語を機械語に翻訳する仕事が必要になる．このアセンブラ言語から機械語への翻訳をアセンブルすると言う．このアセンブルの作業は，コンピューターに任せればかってにやってくれる．そのためのソフトウェアーをアセンブラーと言う．

アセンブラーが開発されたことで，プログラムはかなり容易になったが，それでもまだ，一部の専門家にしか使えない．もっと，人間に近いプログラミング言語の必要性が叫ばれた．そこで，1950 年代初頭，最古の高級言語である FORTRAN が登場した．諸君が学習した通り，誰でもコンピューターのプログラムが書けるようになったのである．例えば，加算は，

```
C=A+B
```

と書けばよい．人間にとって非常に分かりやすい表現である．しかし，コンピューターにとっては全く分からない表現になってしまった．そのために，この FORTRAN 言語で書かれたプログラムを機械語に翻訳する必要がある．翻訳は，コンピューターに任せればよく，そのためのプログラムが用意されている．それは，コンパイラーと言われるプログラムである．皆さんは，FORTRAN コンパイラーを使ったことがある．C 言語のコンパイラーを使ったことがある人もいるであろう．

コンピューターのプログラム言語は，誰でもが間単にプログラムできるように進化してきた．その進化は今でも留まっておらず，いろいろな言語が開発されている．

5　高級言語の実行方法

FORTRAN や C 言語，BASIC，Java，COBOL 等の人間のわかりやすいプログラム言語は，高級言語と呼ばれている．一方，アセンブラ言語や機械語のように，コンピューターが理解できる言語は，低級言語 (?) と呼ばれている．

高級言語といっても，コンパイラーを使って，最終的には機械語に翻訳していることを忘れてはならない．どのような，高級言語でも，コンピューター上で実行する場合，最終的には機械語に翻訳する必要がある．機械語の 0 と 1 の塊が，メモリーにロード されて，それが CPU の論理回路で処理されるのである．

6　アセンブラ言語を学習する理由

コンピュータープログラムは誰でも，使えるように機械語から高級言語に進化してきた．それでは，今，アセンブラ言語を学習する意味があるのか?．アセンブラ言語を勉強する 3 つの理由を示して，本日の講義は終わりとする．

明らかに，アセンブラ言語が高級言語に勝っている点がある．一つは実行速度である．高級言語をコンパイルして，機械語にすると，どうしても処理が複雑になる．たとえば，エラーの処理とか，コンパイラーがかってにルーチンを追加する．アセンブラ言語でプログラムするとそのような処理は不要であれば記述しなければ良いので，機械語が簡単になる．機械語が簡単であれば，それだけ速度が早くなる．

2 つ目の理由は，ハードウェアーに近いところで，細かい制御が可能ということである．特定の制御線の電圧を 1 や 0 に変えることが，アセンブラで直接記述できる．

諸君がアセンブラ言語を学習する最も大きな理由は，コンピューターの仕組みを理解するにはちょうど良いからである．アセンブラ言語は，機械語と 1 対 1 に対応している．機械語の動作は，そのコンピューターのハードウェアー (CPU) の動作と考えても差し支えない．そのため，アセンブラ言語を理解するとコンピューターの動作や原理が理解できるのである．がんばって理解しよう．

7 課題 (レポート)

7.1 コンピューターの基礎的な内容

[**問 1**] ノイマンアーキテクチャーとハーバードアーキテクチャーを調べ, それらの違いについて記述せよ．

[**問 2**] 3.1 節で示したプログラムを真似て，58×28 を計算するプログラムを作成せよ．

[**問 3**] 3.1 節で示したプログラムを真似て，$58 \div 3$ の商と余りを計算するプログラムを作成せよ．引き算を繰り返すことにより、割り算の演算を実現すること．

7.2 レポート 提出要領

提出方法は，次の通りとする．

期限	10 月 21 日 (金)AM 10:35 まで
用紙	A4
提出場所	山本研究室の入口のポスト．授業開始前の手渡しは受け付けない．
表紙	表紙を 1 枚つけて，以下の項目を分かりやすく記述すること． 授業科目名「電子計算機」 課題名「課題 1 コンピューターの基礎」 3E　学籍番号　氏名 提出日
内容	きちんとした文章 (英語または日本語) で，分かりやすく記述すること．
注意	他人のものを写した者，他人に写させた者のレポートはゼロ点とする． 期限厳守．1 秒でも遅れたら、受け取らない． 分からない部分があったも提出すること．ただし、3 時間程度は, 調べ考えること．10 分程度しか考えないで，解答がかかれていないレポートの場合、点数を非常に低くする．