コンピューターとは、どんな機械だろうか?

山本昌志*

2004 年 2 月 13 日

1　本日の授業の内容と目標

本日は、コンピューターの動作の基本的なところを学習する。学習内容とそれで修得すべき目標を示す。
[内容]

- コンピューターの理論モデルとして、チューリング機械を示す。
- ノイマン型コンピューターを説明します。
- COMET II とチューリング機械、およびノイマン型コンピューターの関係を示します。
- アセンブラを通して、コンピューターの動作を考えます。

[目標]

- コンピューターとチューリング機械の対応が分かり、動作原理が理解できる。

2　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」を参考にしています。

2.1　チューリング機械

コンピューターとは、「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 1 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。

*国立秋田工業高等専門学校　電気工学科

- テープには、いろいろな記号が書かれている。
- 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。

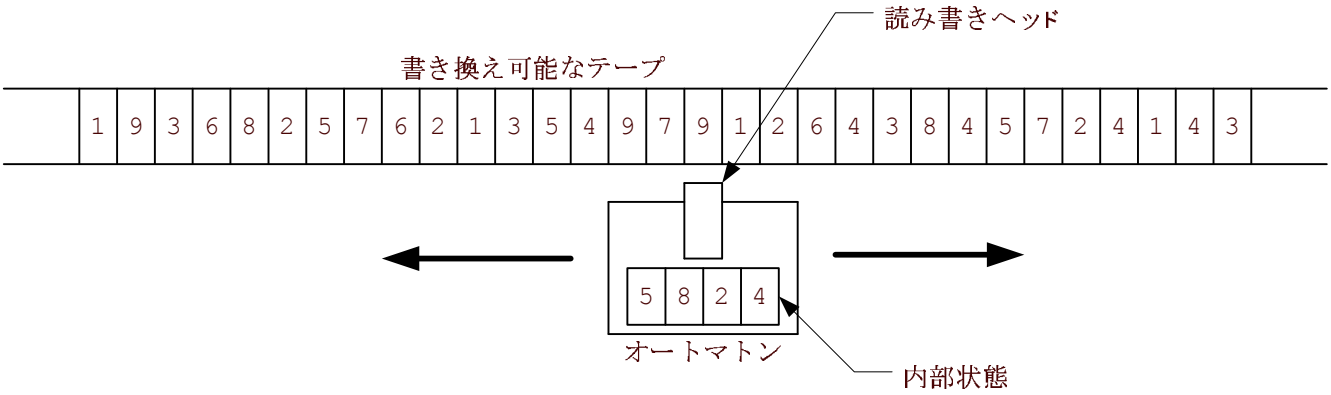

図 1: チューリング機械

これが、コンピューターそのものであると理解できる人は、よく勉強してきた人です。書き換え可能なテープはメモリーに、オートマトンは CPU に相当します。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します。内部状態はレジスタの値に対応します。

あるときチューリング機械が、図 1 の状態であったとします。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ります。次の動作を決めるのは、テープの内容と内部状態により一意に決まります。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す。

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しょうか?。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す。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ます。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す。要するに、チューリング機械ではこの種の計算が、可能なわけです。ただ、チューリング機械で計算できないものもあります。この問題は、込み入って複雑なので、ここでは取り扱いません。

チューリング機械の概要が分かったと思います。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す。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ことは驚きです。

2.2 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ます。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、電卓と同じようなんもので

す。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です。これでは、とても高速にプログラムが実行できるとはいえません。

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です。ドイツの Z3 とハーバード大学のマーク I である。これらの機械は、計算はリレーで行われました。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ました。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ます。これは、高速に計算する上で非常に大きな問題でした。

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ました。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したが、大きな問題がありました。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ありました。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すが、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す。これにより、高速に計算ができたわけです。

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ました。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出来上がります。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われるようになりました。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、定かではありません。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ようです。

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りました。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います。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。

の特徴をもっています。チューリング機械そのものです。

2.3 COMET II のモデル

COMET II は非常に簡単なコンピューターです。もちろん、チューリング機械であるし、ノイマン型コンピューターになっています。もう一度、COMET II のモデルは、図 2 のようになっています[1]

それでは、チューリング機械と COMET II の対応を考えましょう。それは次のようになります。

[1] このモデルのもう少し詳しい説明は、以前の講義ノート
http://www.ipc.akita-nct.ac.jp/~yamamoto/lecture/2003/3E/lecture_3E/computer_shikumi.pdf
を参照のこと。

書き換え可能な紙テープ	メモリー
読み書きヘッド	データバス
内部状態	レジスターの値
オートマトンの移動	データバスの変化

このことから、COMET II はチューリング機械であることが分かります。ということは計算ができることになります。

さらに、ノイマン型コンピューターの最初の特徴、「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」もすぐに分かります。2 番目の特徴「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」も、チューリング機械であればそうなっているはずですが、これは次の章の実際のプログラムで確認します。

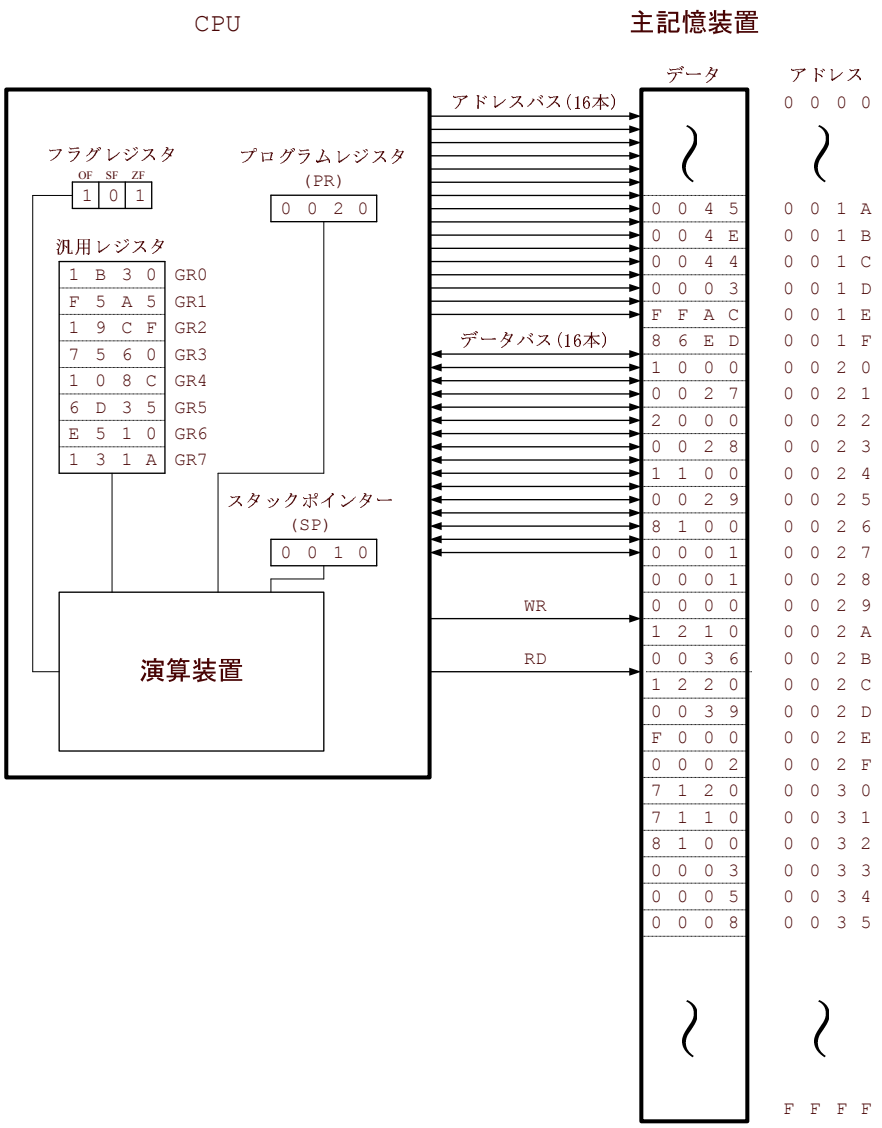

図 2: COMET II のモデル

3　アセンブラを通してのコンピューターの動作

ここで示しているプログラムは、先週の実習で動かしたプログラムです。もう一度、チューリング機械あるいはノイマン型コンピューターとして、このプログラムの意味を考えます。この辺ことを理解するには、高級言語よりも、アセンブラ言語で考えるのが適切です。アセンブラ言語は、ハードウェアーと完全に 1 対 1 に対応しているからです。

3.1　データと命令の区別がない例

次のプログラムは、1+1 を計算する典型的なプログラムです。最後の実行文' C DC 0 ' が' C DS 1 ' と書かなくてはならないと考える人がいると思います。チューリング機械、あるいはノイマン型コンピューターなので、それはどちらでも同じであることが分かると思います。このアドレス C には計算結果を格納するので、' C　 DC 0 ' で 0 に初期化して 1 ワード確保しても、' C DS 1 ' で初期化無しで 1 ワード確保しても同じです。

このプログラムは、命令とデータが区別されています。命令は'LD GR0,A' から、' RET' までです。データは、ラベル A, B,C の部分です。命令とデータが区別できるのはプログラマだけです。コンピューターにとっては、これをアセンブルした機械語しか分からないので、どれが命令でどれがデータかは分かりません。コンピューターにとって、命令であろうとデータであろうとただの 1 と 0 のビットの集まりにすぎません。後は、チューリング機械のように、それに従い動くだけです。

```
PGM   START
      LD     GR0,A
      ADDA   GR0,B
      ST     GR0,C
      RET
A     DC     1
B     DC     1
C     DC     0
      END
```

次のプログラムを見てください[2]。これは、プログラマから見るとデータの集まりです。しかし、コンピューターから見ると全く異なって見えます。この辺の事情は先週のプログラムの実習で体験したとおり、先のアセンブラのプログラムと全く同じです。さきのアセンブラのプログラムを機械語にしたものを、ここでは DC 命令でメモリー上に書き込んでいるからです。

以上で、ノイマン型のコンピューターは命令とデータの区別が無いことが分かったと思います。

```
PGM   START  NEXT
NEXT  DC     #1000
      DC     #0027
      DC     #2000
      DC     #0028
      DC     #1100
```

[2]このプログラムは#0020 からメモリーにロードされ、そこから実行されると仮定している。

```
        DC      #0029
        DC      #8100
        DC      #0001
        DC      #0001
        DC      #0000
        END
```

3.2 命令を作りながらの実行

これは、もう少し野心的なプログラム[3]で、プログラム自身が命令を書き換えて、それを実行するようになっています。これも、チューリング機械だからこそできる芸当です。このプログラムの主な動作は、

- プログラムの動作は、ラベル A の値と B の値の和を C に格納するものである。
- プログラムの実行命令は、ラベルの ONE と TWO に書かれる。
- 最初、ラベル ONE と TWO にには何もしない命令 NOP が書かれている。プログラム実行時には、TWO の後の LAD 命令から JUMP 前までの ST 命令で、ONE と TWO に実効命令が書かれる。
- JUMP 命令で、ラベル ONE の命令が実行される。

です。要するに、ラベル ONE と TWO の命令が実行されるのですが、この命令はこのプログラム自身が作っています。プログラムを作りながらプログラムを実行していることになっています。

```
PGM     START
        LAD     GR1,-1
ONE     NOP
TWO     NOP
        LAD     GR1,1,GR1
        LD      GR2,P,GR1
        ST      GR2,ONE
        LAD     GR1,1,GR1
        LD      GR2,P,GR1
        ST      GR2,TWO
        JUMP    ONE
A       DC      1
B       DC      1
C       DC      0
P       DC      #1000
        DC      #0032
        DC      #2000
        DC      #0033
        DC      #1100
        DC      #0034
        DC      #8100
        END
```

[3]このプログラムは#0020 からメモリーにロードされ、そこから実行されると仮定している。