COMET IIを通して学ぶコンピューターの仕組み

山本昌志*

2004年7月6日

1　先週の復習と本日の学習

1.1　先週の復習

先週は、プログラムの命令をビットパターンに変換する方法を学習した。その方法は単純で、

- アセンブラの命令のビットパターンは、教科書 p.213 の命令語の構成の通りである。
  - 命令にアドレス (通常はラベル名) が含まれない場合、マシン語は 1 ワード (16 ビット) で構成される。
    - 上位の 8 ビット (16 進数の 2 桁) が命令の種類を表す。
    - 下位の 8 ビットが、処理の対象の汎用レジスターを表す。
  - 命令にアドレスが含まれる場合、マシン語は 2 ワードで構成される。
    - 第一語の上位の 8 ビットが命令の種類を表す。
    - 第一語の次の 4 ビット (第 4〜7 ビット) が、処理の対象の汎用レジスターを表す。
    - 第一語の最下位の 4 ビット (第 0〜3 ビット) が、処理の対象の指標レジスターを表す。指標レジスターが無い場合は、$(0)_{16}$ となる。
    - 第二語は、処理の対象のアドレスを表す。

とすればよい。この命令コードとビットパターンの変換表のことをオペレーションコード (operation code)、略して OP コードと言う。

これで、整数と文字、そして命令までビットパターンで表せたことになる。実際のコンピューターでは、このビットパターンが線路の電圧のパターンとなる。そのパターンに従い、動く自動機械に過ぎない。この自動機械の設計には、2 年生の時に学習したブール代数が役に立つ。これまでの学習で、諸君はほとんどコンピューターの基礎的なことを学習したわけである。もし、いままで学習した知識を全て持って、1940 年にワープできたら、世界的なコンピューターの権威になれるはずである。

*国立秋田工業高等専門学校　電気工学科

1.2 今週の学習内容

いままでは、ハードウェアー寄りの話をした。特に重要なことは、命令でもデータでもビットパターンで表現することである。ここではもう少し、掘り下げて、そのビットパターンが電圧を表し、それがコンピューターのハードウェアーになっていることを示す。ビットパターンというソフトウェアーが、電圧というハードウェアーに出会うところである。

2 COMET IIのハードウェアー

アセンブラ言語を学習するためには、コンピューターの仕組みを理解する必要がある。そこで、もう一度コンピューターの仕組みを説明する。これは、アセンブラ言語を学ぶもっとも基本的なことである。どんな学問でも基本さえしっかり理解していれば、後はなんとかなる。

コンピューターの仕組みはいたって単純である。それを、COMET II の Central Processing Unit(CPU) と主記憶装置 (メインメモリーとも言う) だけを考えることにより理解しよう。どんな、コンピューターでもこの 2 つは必ず有る。キーボードが無いとかディスプレイが無いとか、ハードディスクが無いコンピューターは存在するが、CPU あるいは主記憶装置の無いものは存在しない。どんなコンピューターでもその基本的な仕組みは同じである。スーパーコンピューターであろうが COMET II であろうが同じ仕組みで動作している。まずは、それぞれの役割を、もう一度、示す。

図**??**を見ながら、以下の COMET II の構成機器の動作内容を理解すること。

2.1 CPU

CPU の役割は、命令に従いデータの加工 (演算) を行うことである。その命令は、単純である。単純なことしかできないが、その処理速度は、信じられないくらい高速[1]である。CPU の回路は、2 年生の時に学習した論理回路 (組み合わせ回路、順序回路[2]) で構成されている。そこで、論理回路はどんなの入出力の論理でも可能であることを学習したはずである。それも、たった 3 つ (and, or,not) の回路の組み合わせで、できるから驚きである。このことから、どのような処理でも可能な回路ができることが分かる。

2 年生の時、or(論理和) と and(論理積)、not(否定) の回路がトランジスターで出来ることを学習した。ブール代数というソフトウェアーがトランジスターというハードウェアーで実現できるのである。コンピューターはまさにこれである。すなわち、トランジスターがビットパターン (命令とデータ) 応じた電圧を制御することにより、論理演算を行っている。要するに、いままで学習したビットパターンは、コンピューター内部では電圧のパターンに変換されて、トランジスターにより論理演算を行うのである。論理演算を行う装置は、以前学習した加算器のようなものである。

COMET II の場合、扱うデータは全て 16 ビットである。この 16 ビットのデータを処理することが、CPU の役割である。

単純な演算を高速に処理することは得意であるが、記憶は苦手である。記憶容量は小さく、図**??**に示すように 10 個程度のレジスタに演算に必要な情報を記憶する。それぞれのレジスタは役割が決まっており、次の役割を担っている。

[1] 例えば、Intel 社の Pentium の場合、3GHz で動作する。1 秒間に 30 億回、何かをするのである。
[2] 順序回路は 4 年生で学習する。

プログラムレジスタ (PR) 次に実行する命令のアドレスを示す。そのため、実行した命令が 2 語で構成されていれば、+2 加算される。1 語であれば、+1 加算される。

汎用レジスタ (GR0〜GR7) 演算を行うためのデータを格納する。GR1〜GR7 は、指標レジスター (index register) としても使われる。

フラグレジスタ (FR) 演算結果の状態を示す。演算結果の状態とは、**OF:**オーバーフローの有無、**SF:**正負、**ZF:**0 か否かである。

スタックポインター (SP) スタック領域の最上位のアドレスを示す。これは、かなり後で学習する。

2.2 主記憶装置 (メインメモリー)

主記憶装置の役割は、命令や処理をするデータ (数字や文字) を記憶することである。記憶する場所は 65536 個有り、それぞれに整数の番号が割り振られている。その番号を番地と言い、通常 16 進数で表し、COMET II の場合、その範囲は`#0000`〜`#FFFF` 番地である。

主記憶装置の番地は 16 ビットの整数で表現され、また、そこに格納される内容も 16 ビットである。CPU のデータの受け渡しは、CPU の命令に従い、次に説明するバスを通して行われる。

2.3 バス

バス (bus) とは、コンピュータ内部で各回路がデータをやり取りするための伝送路のことである。ここでは、CPU と主記憶装置をつなぐ、電線のことを言う。COMET II や CASL II の仕様 (教科書 p.207〜214) には記述されていないが、コンピューターを考える場合、CPU と主記憶装置をつなぐ、電線は必要であろう。この仕様から、少なくとも、以下の電線が必要なことはわかるであろう。

アドレスバス CPU が主記憶装置の記憶内容を読み書きする場合、その場所を示すための線である。アドレスは 16 ビットなので、当然その線の数も 16 本である。この 16 本の線を 0(Low) と 1(High) にすることにより、主記憶装置のアドレスを指定する。この線を通しての情報の流れは、CPU から主記憶装置のみで、一方的である。主記憶装置が CPU にアドレスを指定することは絶対にないのである。

データバス CPU とメモリーとの間で、データを受け渡しするための線である。COMET II のデータは全ていつも 16 ビットなので、16 本の線が必要である。情報の流れは、双方向である。

WR 線 CPU が主記憶装置に内容を書き込む場合、CPU がこの線を 1(High) にする。この線を通しての「主記憶装置にデータを書き込む」という情報の流れは、CPU から主記憶装置のみで、一方的である。

RD 線 CPU が主記憶装置から内容を読み込む場合、CPU がこの線を 1(High) にする。この線を通しての「主記憶装置から内容を読み込む」という情報の流れは、CPU から主記憶装置のみで、一方的である。

3 COMET IIの動作

次に COMET II の動作の仕組みを考える。COMET II に限らず、どんなコンピューターでも基本的には同じである。

図**??**の状態に主記憶装置がなっていたとする。さらに、プログラムレジスタ (PR) の値が#0020 となっていたとする。主記憶装置のアドレス#0020 以降は、以下の CASL II のプログラムをアセンブルして機械語に直したものである。

```
    PGM   START
1         LD    GR0,A
2         ADDA  GR0,B
3         ST    GR0,C
4         RET
5   A     DC    1
6   B     DC    1
7   C     DC    0
8         END
```

図 1: 1+1 を計算するプログラム

ひとつずつ、このコンピューターの動作を考える。すると、以下のように動作する。

1. プログラムレジスタの中身が#0020 なので、CPU がアドレスバスを (0000 0000 0010 0000) とする。さらに、CPU は RD=1 とする。すると、主記憶装置が、データバスを (0001 0000 0000 0000) とする。
2. CPU が受け取った#1000 を解析する。解析の結果、それは、2 語の命令と分かるので、アドレスバスとデータバス、RD 線を使って、主記憶装置から残りの 1 語分#0027 を読み出す。
3. 2 語読み出したので、プログラムレジスタを+2 加算する。即ち、PR=#0022 となる。
4. 読み出した命令 (#1000 と#0027) から、#0027 番地のデータを読み出して、汎用レジスタ GR0 にその値を入れる。GR0=#0001 となる。そして、フラグレジスタをセット (OF=0,SF=0,ZF=0) する。これで最初の命令が完了。
5. 次に、プログラムレジスタ PR=#0022 に従い、その番地の命令#2000 を読み出す。
6. この命令を CPU が解析して、2 語と分かる。そして、#0023 番地のデータ#0028 を主記憶装置から読み出す。
7. 2 語読み出したので、プログラムレジスタを+2 加算する。即ち、PR=#0024 となる。
8. 読み出した命令 (#2000 と#0028) から、#0028 番地のデータを読み出して、汎用レジスタ GR0 との和を計算して、結果を汎用レジスタに戻す。GR0=#0002 となる。そして、フラグレジスタをセット (OF=0,SF=0,ZF=0) する。これで 2 番目の命令が完了。
9. プログラムレジスタ PR=#0024 に従い、その番地の命令#1100 を読み出す

10. この命令を CPU が解析して、2 語と分かる。そして、#0025 番地のデータ#0029 を主記憶装置から読み出す。

11. 2 語読み出したので、プログラムレジスタを+2 加算する。即ち、PR=#0026 となる。

12. 読み出した命令 (#1100 と#0029) から、汎用レジスタ GR0 の内容を#0029 番地のデータに書き込む。これで 3 番目の命令は終了。

13. プログラムレジスタ PR=#0026 に従い、その番地の命令#8100 を読み出す。

14. この命令を CPU が解析して、RET 命令と分かる。スタックポインター (PR) が示すアドレスの値をプログラムレジスタ PR にセットする (この辺は後の学習範囲)。これでこの命令は終わり。

これで、命令もデータ (整数や文字) が 2 進数で表すか分かるであろう。それは、全てハードウェアーの電圧に対応しているのである。その電圧に対応して、コンピューターは動作しているに過ぎないのである。

4 命令とデータの区別について

主記憶装置に格納されているデータは、16 ビットのただのビットパターンである。16 個の 0 と 1 の集まりにすぎない。16 進数で書くと、4 桁の数字である。その 4 桁の 16 進数の数字が、整数や文字、あるいは命令を表したりする。CPU は、それらをどのように区別しているのであろうか?。そのからくりは?。それらを全く区別していないというのが答えである。ただ単に、プログラムレジスタ PR が示すアドレスの内容は命令と解釈するだけである。すごーく、単純である。後は、その命令に従い、主記憶装置の内容が命令になったり、整数になったり、文字になったりしているだけである。メモリーの内容を見ただけでは、それが示すものは、文字なのか整数なのか、命令なのかはは分からないのである。

COMET II では、命令と処理すべきデータ (整数や文字) が同じところに、区別無く格納されている。世界中にある普通のコンピューターも同じようになっている。このように、命令とデータ区別しないで、同じメモリーに格納することはノイマン型コンピューターの特徴のひとつである。

通常のコンピューターは、とてつもないビットの操作をしていることが分かるであろう。それもひとつも間違えないで行うのは奇跡に等しいと思える。どのようにしているのであろうか?。

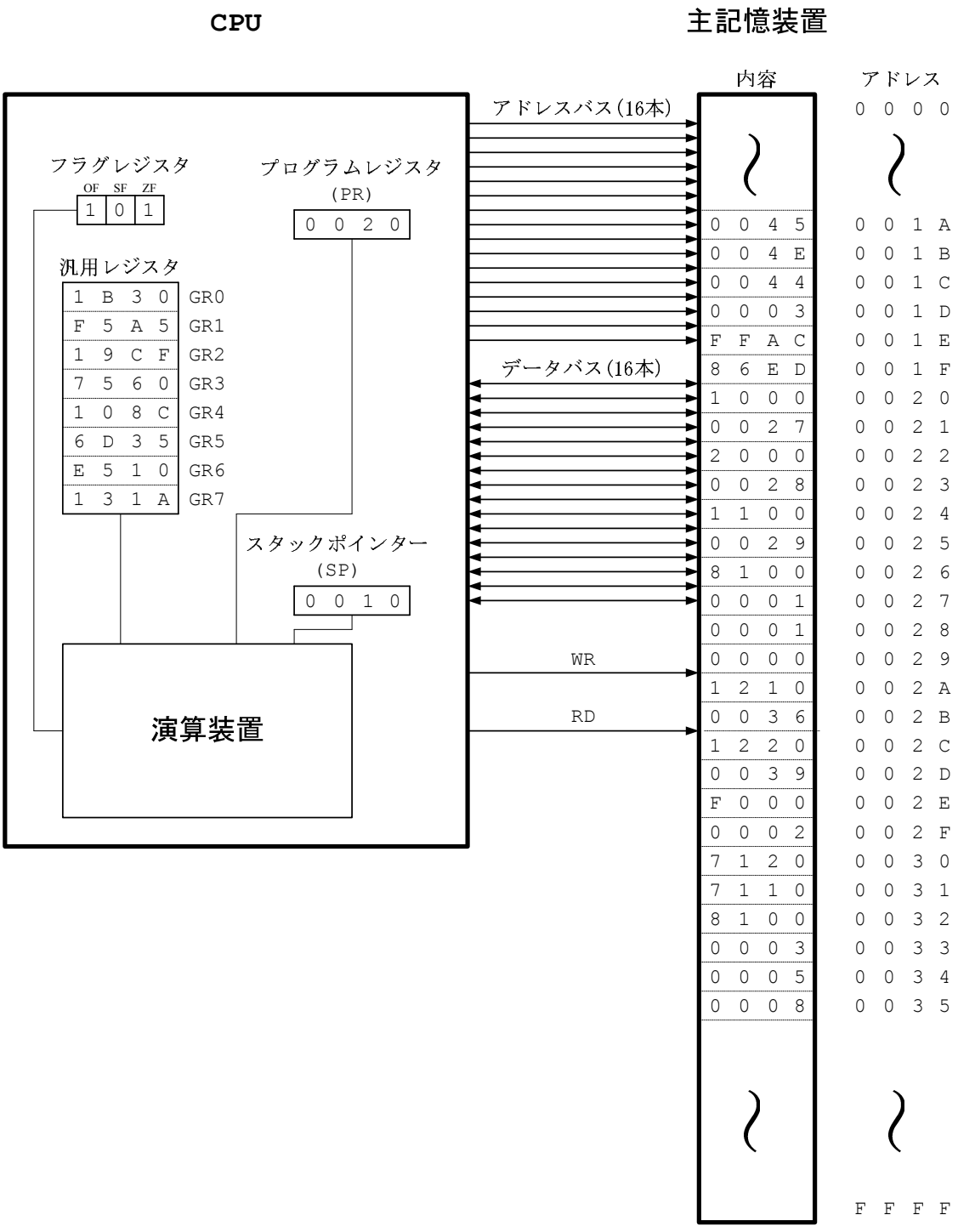

図 2: COMET II のハードウェアーとプログラムの状態