電子計算機 **3E**

3E　2003.8.25

本節の授業のテーマ

本日の事業のテーマは、以下のとおりです。

(1)　CASL II の命令形式
(2)　機械語命令
　　LD
　　ST
　　LAD

本日の授業のゴールは、以下のとおり。

- 機械語命令の書き方が分かる。
- データ転送命令が理解できる。
  - LD
  - ST
  - LAD

1. はじめに

アセンブラ命令の学習が終わりましたので、機械語命令に入ります。機械語命令が終われば、マクロ命令です。他の 2 つの命令に比べて、機械語命令は数も多く、その動作も多岐にわたります。なんと言っても、アセンブラのプログラムの中心はこの機械語命令です。プログラムの目的であるデータの加工は、ほとんど機械語命令で実行されます。

機械語命令は数が多いので、しばらくは機械語命令の学習を進めることになります。命令の文法は教科書に書いてあります。プリントではその補助的な説明を行います。

1.1 機械語命令の書き方

教科書に書かれているように、機械語命令の書き方は、オペランドが異なる 5 種類に分類できます。機械語命令の場合、CASL II の 1 行は機械語命令の 1, 2 ワードのマシン語に変換されて、メインメモリーに格納されます。1 ワードは 16 ビットで、2 ワードは 32 ビットです。その 1 あるいは 2 ワードで、命令の種類と対象であるオペランドを示すマシン語になるのです。

機械語命令の書き方は、以下の通りです。


```
[ラベル]  OP      r1,r2          ;注釈が書けます
[ラベル]  OP      r,adr[,x]      ;注釈が書けます
[ラベル]  OP      adr[,x]        ;注釈が書けます
[ラベル]  OP      r              ;注釈が書けます
[ラベル]  OP                     ;注釈が書けます
```

1.2 オペランドの内容

命令の対象となるオペランドの書き方は、教科書に書いてある通りで 5 種類です。機械語命令の書き方を見よ。これらの内、汎用レジスタが r1 と r2, r です。GR0 とか、GR1 と書きます。汎用レジスタの範囲は、GR0 から GR7 までです。

x は指標レジスタです。指標レジスタについては、第 7 回の授業で説明しましたが忘れていると思いますので、再度説明しておきます。プログラムを書いているとき、基準点のアドレスにある値を加算してデータにアクセスしたい場合がしばしば遭遇します。このようなときに指標レジスタを使います。すなわち、オペランド欄に、

```
adr,x
```

と書いた場合です。adr が基準点のアドレスで、x が指標レジスタです。実際にデータが操作される実行アドレスは、adr+x ということになります。adr はつぎに述べる方法でアドレスを指定します。加算する値を格納するのは指標レジスタで汎用レジスタの GR1～GR7 を使います。どうして、GR0 はダメなのでしょうか?。皆さん考えてください。このように、指標レジスタを用いて、アドレスを操作することをアドレス修飾と言います。

adr は、アドレスを示します。アドレスの書き方は後で示します。

1.3 アドレス

COMET II のメインメモリーのアドレスは、16 ビットです。従って、アドレスの範囲は、0～65535(#0000～#FFFF)番地です。このメモリー空間は、20 年くらい前の 8 ビットパソコンと同じです。64k バイトです。ちなみに、いま主流の 32 ビットパソコンのメモリー空間は 32 ビットで、4G バイトになります。メモリーにアクセスする場合、番地を指定します。その番地の指定方法を述べます。

教科書に書かれている通り、アドレスはつぎに示す 3 通りの方法で記述できます。最初の 2 つの 10 進数と 16 進数を使う場合、絶対アドレスを指定することになります。よっぽどのことがないかぎり、絶対番地を指定することはありません。なぜならば、実際のプログラムを実行する場合、データがどの番地に格納されているかは、プログラマは分かりません。プログラム実行段階で、OS が決めるからです。従って、みなさんは最後のアドレス定数を使いましょう。

10 進定数　　10 進数の定数を用います。内容は、教科書に書かれている通りです。

16 進定数　　16 進数の定数を用います。16 進数であることを表すために、先頭に#を付けます。

アドレス定数　ラベル名を指定します。アセンブラーにより、ラベル名がアドレスに変換されます。

1.4 リテラル

機械語命令のオペランドの adr は、アドレスを示すことは先に述べた通りです。アドレスの指定は、10 進定数と 16 進定数、アドレス定数があります。さらに、リテラルでもそれを指定できます。リテラルを用いたアドレスは、10 進定数や 16 進定数、あるいは文字定数の前に‘=’の記号をつけます。

教科のリテラル形式をアセンブルすると、DC 形式のマシン語になります。あとは、教科書の通り。

2. データ転送

メインメモリーからレジスタに、あるいはレジスタからメモリーにデータを転送する命令の使い方を示します。いずれの場合も、1 回のデータの転送量は 1 ワードです。メモリーやレジスタの領域は複数ありますので、その位置を指定する必要があります。メモリーの場合は先ほど示した方法でアドレスを指定します。レジスターの場合はその名前で転送場所を指定します。

2.1 LD

COMET II では、算術演算や論理演算は必ず汎用レジスター上で行われます。そのため、演算の対象となるデータを汎用レジスターに格納する必要があります。LD 命令は、メインメモリーのあるアドレスのデータを汎用レジスタにコピーする命令として使われます。それればかりではなく、汎用レジスター間のコピーにも使われます。しかし、汎用レジスターからメインメモリーや、メインメモリーからメインメモリーへのコピーはできません。メインメモリーへのコピーは ST 命令を使います。

語源は、英語の load です。load のにはいろいろ意味がありますが、その中で`読み込む`と意味で使われています。

文法は教科書に書いてあるとおりです。使い方の例を示します。

例 1 汎用レジスタ GR1 の内容を GR0 にコピーします。

```
LD GR0,GR1
```

例 2 アドレス定数 A で示されるメインメモリーのアドレスのデータを汎用レジスタ GR0 にコピーする。アドレス定数 A の A はラベル名である。

```
LD GR0,A
```

例 3 指標レジスターGR1 を用いて、アドレス修飾されたアドレスを汎用レジスター GR0 にコピーする。

```
LD    GR0,A,GR1
```

レジスタにコピーされた値によって、フラグレジスタの値は変化します。フラグレジスタの各ビットは、常識とおりで教科書にかかれています。この性質を利用して、レジスタの符号を調べることができます。

例 4 GR1 レジスタの符号がフラグレジスタの SF にセットされる。また、ゼロか否かも ZF にセットされる。

```
LD    GR1,GR1
```

2.2 ST

ST 命令は、LD 命令とは逆に、汎用レジスタの内容をメインメモリの指定番地にコピーします。語源は STore で、`備蓄する`などの意味があります。書式や機能は教科書に書かれている通りです。

フラグレジスタは変化しません。

例 1 汎用レジスタ GR0 の内容をアドレス定数(ラベル)A で示されるメインメモリーの番地にコピーする。

```
ST GR0,GR1
```

例 2 アドレス修飾を用いて、GR0 の内容を実行アドレス A+GR1 にコピーする。

```
ST GR0,A,GR1
```

2.3 LAD

LD 命令は、メインメモリーの指定した番地の内容(データ)を汎用レジスタにコピーします。一方、LAD 命令は、その番地(実行アドレス)を汎用レジスタにコピーします。

実行アドレス(adr,[x])の指定に 10 進定数や 16 進定数を指定することにより、直接、汎用レジスタに値を格納することができます。そのため、汎用レジスタに初期値を設定したり、制御変数の値を操作するときに使われ、非常に使い勝手のある命令になっています。

フラグレジスタは変化しません。

例 1 アドレス定数 A のアドレスを汎用レジスタ GR0 にコピーする。

```
LAD GR0,A
```

例 2 アドレス修飾されたアドレスを汎用レジスタ GR0 にコピーする。

```
LAD GR0,A,GR1
```

その他の例については、教科書を参照のこと。