

5.3 アセンブラ

コンピュータが直接理解できるのは**機械語**で、今でも組み込み式コンピュータなど限られた部分ではありますが使用されています。また、コンピュータが登場した当時はこの言語を使ってプログラムを作成していましたが、この言語は直接コンピュータの資源を利用するためプログラムの追加・削除・変更が容易ではありませんでした。その後、機械語と1対1に対応した**アセンブラ言語**が登場し、プログラムを実行する直前に機械語に変換したので、機械語の欠点が改善されました。しかしながら、この言語も中央処理装置の違いによってアセンブラ言語が異なっていて、本テキストで学習するC言語のように汎用性に富んではいませんでした。その後、汎用性に富んだ言語が次々に現れることとなります。

情報処理試験で用いられる仮想のコンピュータを例に挙げ、コンピュータの計算の仕組みを詳しく見ていきましょう。COMET II と呼ばれる5大要素を備えた仮想のコンピュータとその上で動くCASL II と呼ばれるアセンブラ言語を用います。ノイマン型のコンピュータの制御方式は**逐次制御**で「**フェッチ**(命令の取り出し) ⇒ **デコード**(命令の解読) ⇒ **実行**」を順番に繰り返すことによって動作しています。これらを実現するためにコンピュータの基本構成は下図のようになっています。

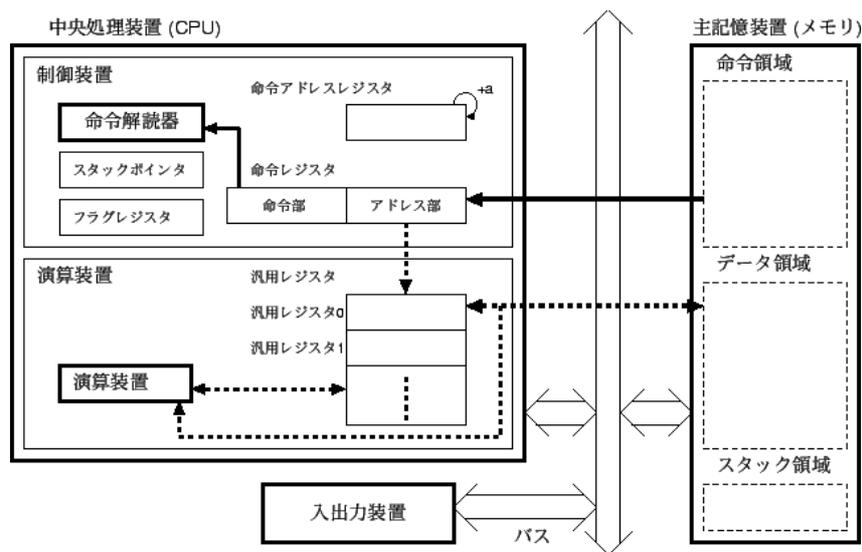


図 5.4: 計算機内部の模式図

主記憶装置 **主記憶装置**はノイマン型のコンピュータの特徴で、命令とデータを記憶するところです。主に命令領域とデータ領域に分れますが、その他にスタック領域として使用します。COMET II では1語は16ビット(2進数で16桁)で表され、アドレスは16ビット(0000(16)からFFFF(16))で表現できる65536語を記憶します。

命令解読器 **命令解読器**は制御装置の中心的存在で命令レジスタの命令部を解読し、各装置に命令を出します。命令デコーダとも呼ばれます。

演算装置 **演算装置**では記憶装置のデータや汎用レジスタの値を用いて計算を行います。また、命令レジスタのアドレスの計算なども行います。

命令アドレスレジスタ **命令アドレスレジスタ**は次にどこからプログラムを進めれば良いのか位置 (アドレス) を記憶しています。命令の語の長さによって加数は変化します。

命令レジスタ **命令レジスタ**は命令を記憶する場所で命令部とアドレス部に分れています。また、命令部は命令解読器に送られアドレス部は演算装置に送られます。

汎用レジスタ **汎用レジスタ**は演算装置に処理するデータや処理されたデータが一時的に記憶される場所です。

スタックポインタ 記憶装置にスタック領域が確保され、**スタックポインタ**はその先頭アドレスを記憶しています。また、スタックには最後に入れたものを最初に取り出す (LIFO:Last In First Out) 機能があります。

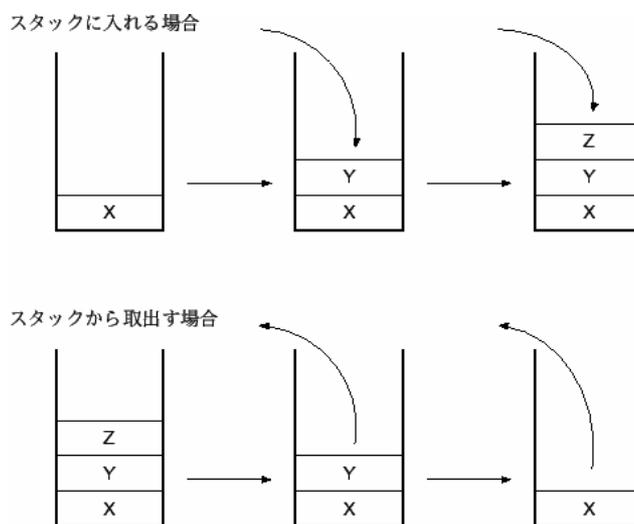


図 5.5: スタックの仕組み

フラグレジスタ **フラグレジスタ**はオーバーフロー (OF), サイン (SF), ゼロ (ZF) フラグで構成され、演算装置による演算結果により値を変えます。

バス **バス**は CPU とメモリに引かれた道路で命令やデータが流れます。

コンピュータのプログラム実行は次の手順によって実行されていきます。

step 0 コンピュータでプログラムが実行される時、最初に命令アドレスレジスタの値が 0 に初期化されます。

step 1(フェチ) 0 番地の命令が命令レジスタに読み込まれます。

step 2(デコード) 命令解読器に送られ命令の解読が行われます。その際、命令レジスタの語長によって命令アドレスレジスタの値を次に実行する命令アドレスまで進めておきます。

step 3(実行) 命令によって各装置を制御します。

step 4 step 1に戻り命令アドレスレジスタが示す命令をフェチします (命令アドレスレジスタは次に実行する命令のアドレスを示しています)。

それではもう少し詳しく知るために主な命令に対して各装置の動作を見ていきましょう。

ロード命令 もっとも基本的な命令でメモリ又は汎用レジスタの値を他の汎用レジスタに代入します。

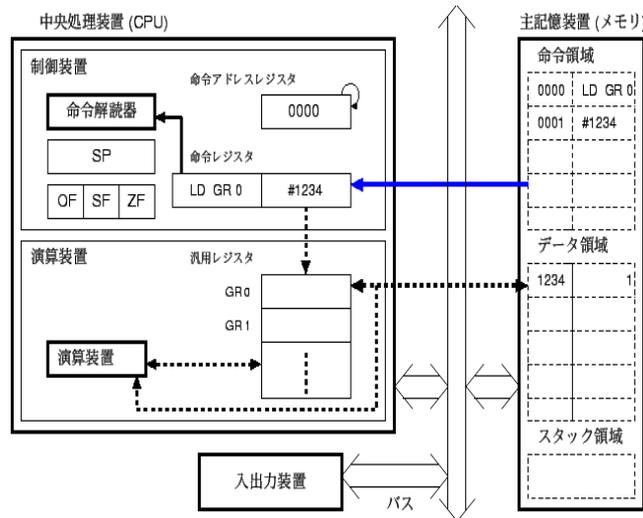


図 5.6: ロード命令:フェチ

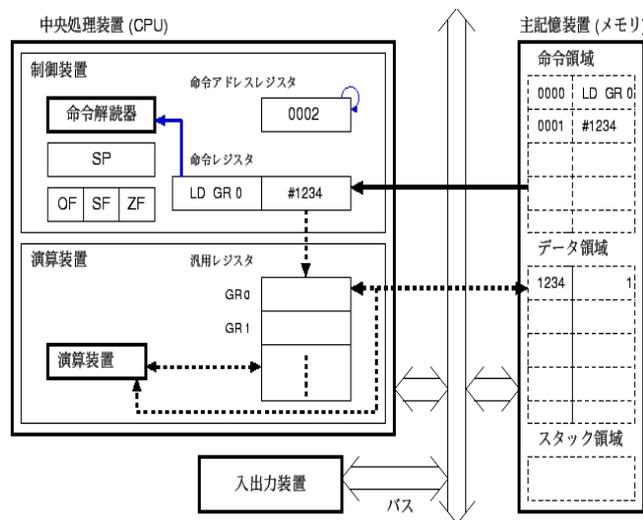


図 5.7: ロード命令:デコード

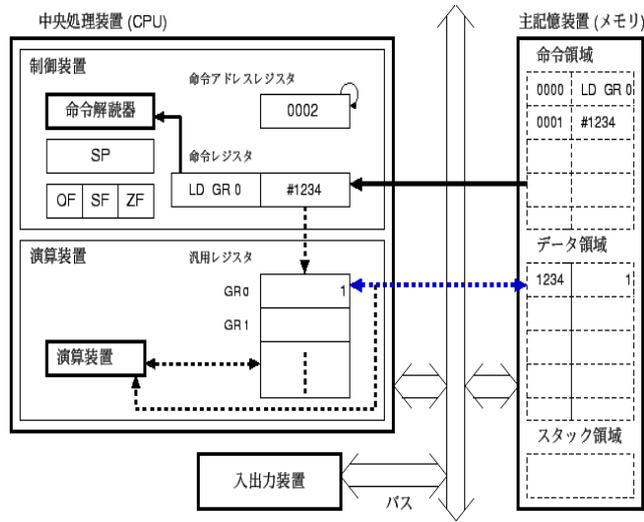


図 5.8: ロード命令:実行

算術加算命令 汎用レジスタにメモリ又は他の汎用レジスタの値を加えます。

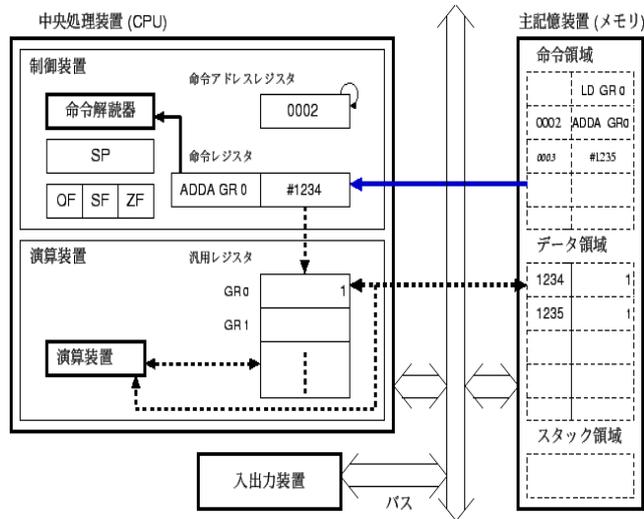


図 5.9: 算術加算命令:フェッチ

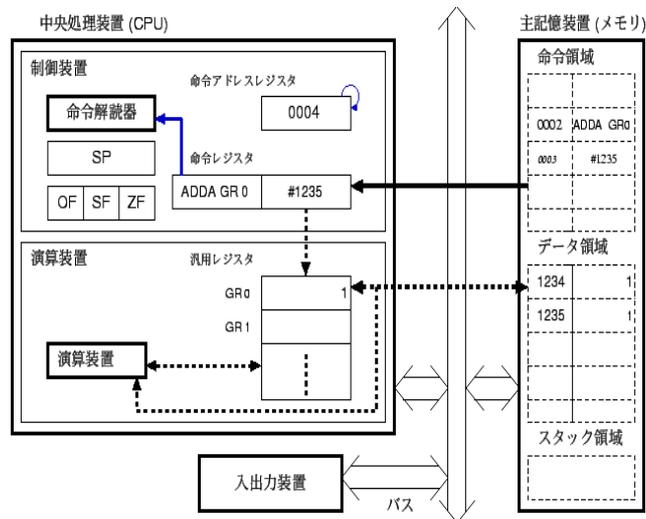


図 5.10: 算術加算命令:デコード

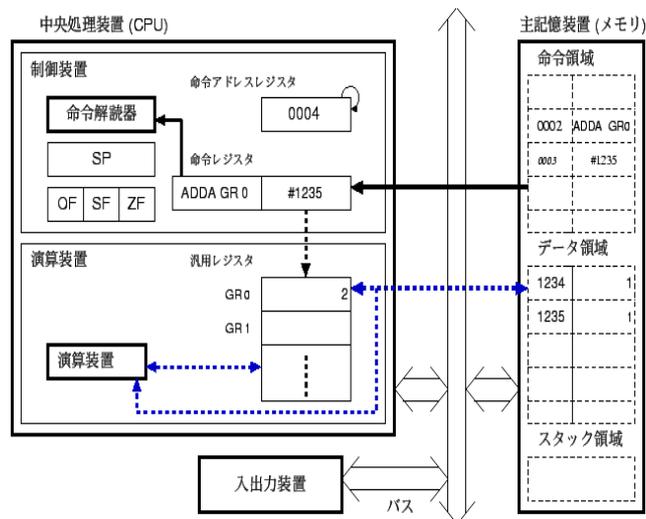


図 5.11: 算術加算命令:実行

比較演算命令と分岐命令 比較演算命令によってフラグレジスタの値が変わったとき続けて分岐命令を行うと条件判別のプログラムとなります。

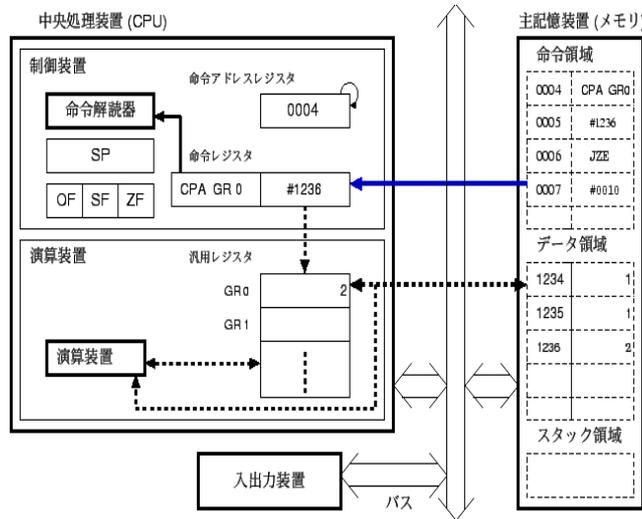


図 5.12: 比較演算命令:フェッチ

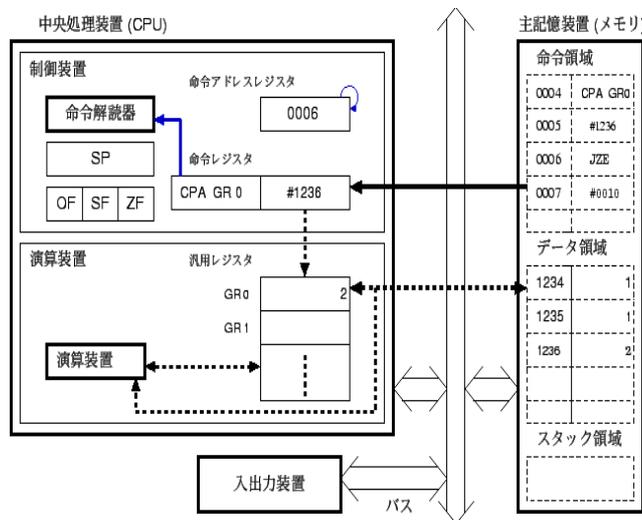


図 5.13: 比較演算命令:デコード

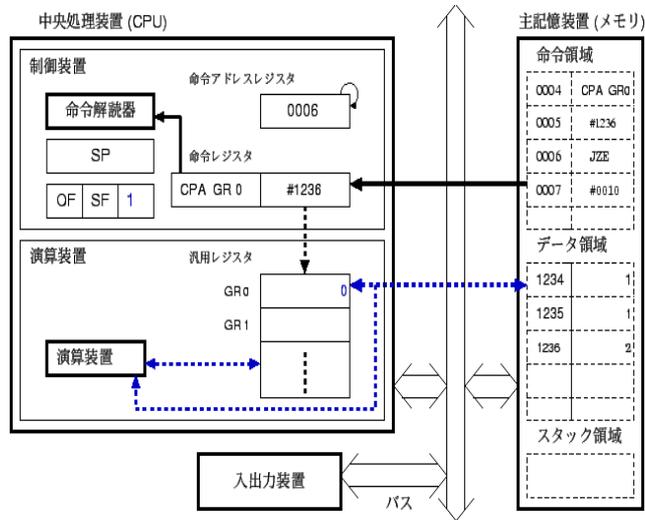


図 5.14: 比較演算命令:実行

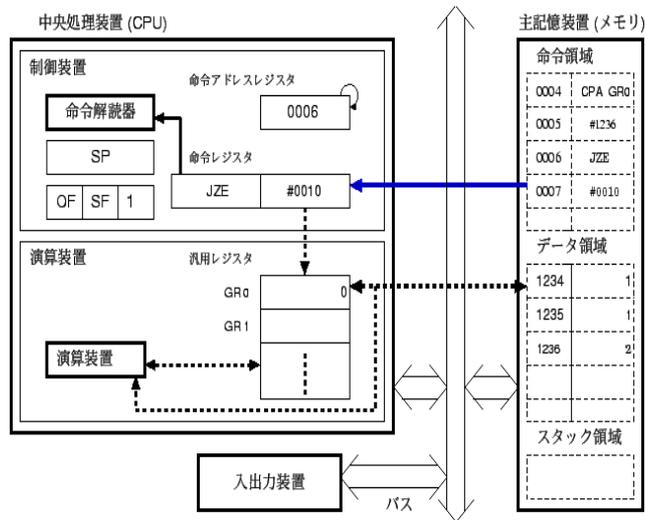


図 5.15: 分岐命令:フェッチ

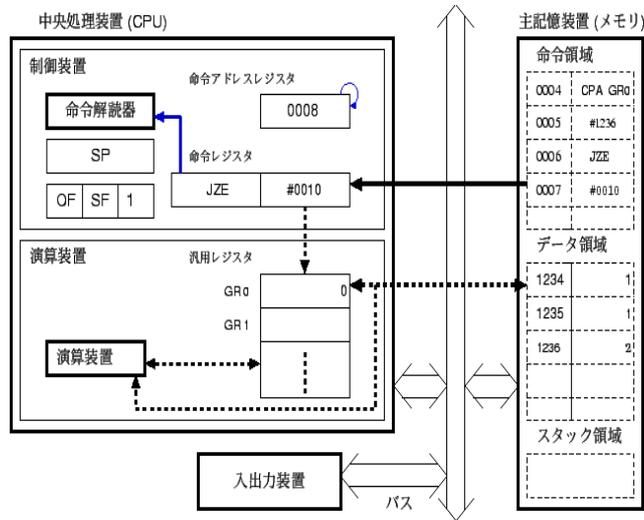


図 5.16: 分岐命令:デコード

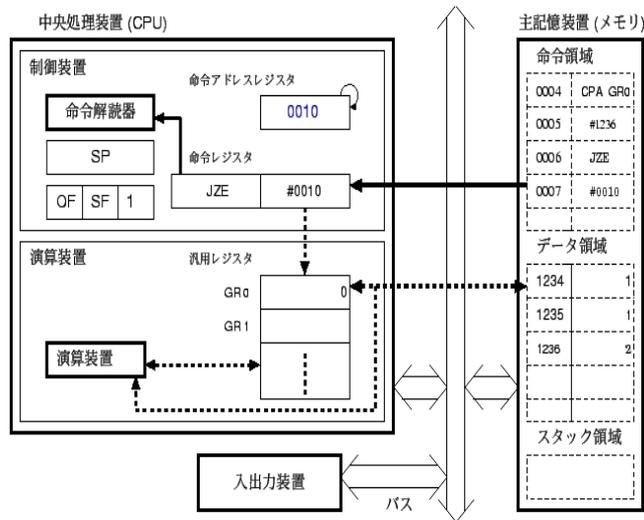


図 5.17: 分岐命令:実行

次のプログラムは1から10まで繰り返し足し算を行うプログラムです。

```
1: PROG1   START
2:         LD     GR0, NUM10
3:         LD     GR1, NUM0
4:         LD     GR2, NUM0
5: LABEL   ADDA   GR1, NUM1
6:         ADDA   GR2, GR1
7:         SUBA   GR0, NUM1
8:         JNZ   LABEL
9:         RET
10: NUM10  DC     10
11: NUM0   DC     0
12: NUM1   DC     1
13:         END
```

正整数 X と Y の積 S を求めるアセンブラプログラムは次のようになります。

```
1: PROG2   START
2:         LD     GR0, NUM0
3:         LD     GR1, NUMX
4:         LD     GR2, NUMY
5:         LD     GR3, NUM0
6: LABEL   LD     GR4, GR2
7:         SRA   GR4, GR3
8:         AND   GR4, NUM1
9:         JZE   NEXT
10:        LD     GR4, GR1
11:        SLA   GR4, GR3
12:        ADDA  GR0, GR4
13: NEXT   ADDA  GR3, NUM1
14:        CPA   GR3, NUM15
15:        JNZ   LABEL
16:        RET
17: NUMX   DC     11
18: NUMY   DC     5
19: NUM0   DC     0
20: NUM1   DC     1
21: NUM15  DC     15
22:        END
```

例として $X=11(10)=1011(2)$, $Y=5(10)=101(2)$ としてプログラムを実行すると次のようになります。

	GR0	GR1	GR2	GR3	GR4	FR
1						
2	0					
3	0	1011				
4	0	1011	101			
5	0	1011	101	0		
6	0	1011	101	0	101	
7	0	1011	101	0	101	00
8	0	1011	101	0	1	00
9	0	1011	101	0	1	00
10	0	1011	101	0	1011	00
11	0	1011	101	0	1011	00
12	1011	1011	101	0	1011	00
13	1011	1011	101	1	1011	00
14	1011	1011	101	1	1011	10
15	1011	1011	101	1	1011	10
6	1011	1011	101	1	101	10
7	1011	1011	101	1	10	00
8	1011	1011	101	1	0	01
9	1011	1011	101	1	0	01
13	1011	1011	101	10	0	01
14	1011	1011	101	10	0	10
15	1011	1011	101	10	0	10
6	1011	1011	101	10	101	10
7	1011	1011	101	10	1	00
8	1011	1011	101	10	1	00
9	1011	1011	101	10	1	00
10	1011	1011	101	10	1011	00
11	1011	1011	101	10	101100	00
12	110111	1011	101	10	101100	00
13	110111	1011	101	11	101100	00
14	110111	1011	101	11	101100	10
15	110111	1011	101	11	101100	10
⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	110111	1011	101	1111	0	01
16	110111	1011	101	1111	0	01

正整数 X と Y の最大公約数 Z を求めるアセンブラプログラムは次のようになります。ただし、 $X > Y$ とする。

```
1: PROG3   START
2:         LD      GR0, X
3:         LD      GR1, Y
4: LABEL1  SUBA    GR0, GR1
5:         JZE     LABEL2
6:         CPA     GR0, GR1
7:         JPZ     LABEL1
8:         LD      GR2, GR1
9:         LD      GR1, GR0
10:        LD      GR0, GR2
11:        JMP     LABEL1
12: LABEL2  RET
13: NUMX   DC      81
14: NUMY   DC      15
15:        END
```

例として $X=81(10)$, $Y=15(10)$ としてプログラムを実行すると次のようになります。

	GR0	GR1	GR2	FR
1				
2	81			
3	81	15		
4	66	15		00
5	66	15		00
6	66	15		00
7	66	15		00
4	51	15		00
5	51	15		00
6	51	15		00
7	51	15		00
4	66	15		00
⋮	⋮	⋮	⋮	⋮
5	6	15		00
6	6	15		10
7	6	15		10
8	6	15	15	10
9	6	6	15	10
10	15	6	15	10
11	15	6	15	10
4	9	6	15	00
5	9	6	15	00
6	9	6	15	00
7	9	6	15	00
4	3	6	15	00
5	3	6	15	00
6	3	6	15	10
7	3	6	15	10
8	3	6	6	10
9	3	3	6	10
10	6	3	6	10
11	6	3	6	10
4	3	3	6	00
5	3	3	6	00
6	3	3	6	00
7	3	3	6	00
4	0	3	6	01
5	0	3	6	01
12	0	3	6	01

問題 1 掛け算のプログラムに習って、正整数 X と Y の商 Z を求めるアセンブラプログラムを作成しなさい。ただし、小数点以下は切り捨てる。

問題 2 5の平方根を求めるアセンブラプログラムを作成しなさい。50000から223を求めればよい。