

第1章 構造化命令とは

1.1 構造化アセンブラ

アセンブラ YAH8 は構造化アセンブラです。構造化アセンブラはアセンブラに高級言語で使われる IF や WHILE などの制御構造を取り入れたアセンブラです。

```
?IF (ER0 >= 9)
    ADD    #-9,ER0
?ENDI
```

制御構造に使われる命令を構造化命令と言います。この構造化命令をアセンブラに取り入れることにより、プログラムからラベルが減り、見やすくすっきりしたプログラムが作成できます。

1.2 YAH8 の構造化命令の特徴

YAH8 の構造化命令は次のような特徴があります。

- 高級言語と同様な複雑な条件式が書ける
- プロシージャ命令により C 言語とのインターフェースが容易
- ショートレンジの構造化命令によってプログラムの効率化が図れる
- 多彩で融通性のある文法
- レジスタを破壊しない

構造化アセンブラの中には例えば、「MOV R0,R1」を「R1 = R0」と書けるようなものもあります。しかしこれは記法の簡略化にしかすぎず、「構造化」とは関係ありません。YAH8 の場合はこのような簡略化の記述まではできませんが、構造化命令を取り入れることにより、プログラムの構造がすっきりすることは間違いありません。

構造化命令は拡張命令ですから、これらの命令を必ず使わなければならないということはありません。使用したくない場合は通常のアセンブラとして使用できます。

1.3 予約語

構造化命令のために次の予約語が新たに設けられました。構造化命令の予約語の先頭には必ず '?' が付きます。

?PROC	?LOCAL	?ENDP	?CALL	?RETURN
?SIGNED	?IF	?SIF	?ELSE	?ELSEIF
?ENDI	?SWITCH	?SSWITCH	?CASE	?DEFAULT
?BREAK	?ENDS	?WHILE	?SWHILE	?ENDW
?REPEAT	?SREPEAT	?UNTIL	?FOR	?SFOR
?ENDF	?CONTINUE	?TRUE	?FALSE	?FLAG_C
?FLAG_V	?FLAG_Z	?FLAG_N		

1.4 構造化命令の概要

IF 文

```
?SIF (ER1 != 0)
    ?SIF (ER0 == 0)
        MOV.L  ER1, ER0
    ?ELSE
        XOR.B  R2L, R2L
        JSR   @fadd
    ?ENDI
?ENDI
```

WHILE 文

```
?SWHILE(ADD.L ER0, ER0, ! ?FLAG_C)
    DEC.W      #1, R2
?ENDW
```

REPEAT 文

```
?REPEAT
    MOV.W      @DATA,R0
    DEC.W      #1, R1
?UNTIL (R0 == 0 && R1 > 0)
```

SWITCH 文

```
?SSWITCH (R4L)
?CASE 3:
    MOV.L      #1, ER4
    ADD.L      ER1, ER0
    ?BREAK
?CASE 0:
    XOR.L      ER4, ER4
    ADD.L      ER1, ER0
    ?BREAK
?CASE 1:
    XOR.L      ER4, ER4
    SUB.L      ER1, ER0
    ?SIF (?FLAG_N)
        NEG.L      ER0
        BNOT      #0, R4L
    ?ENDI
    ?BREAK
?ENDS
```