HY-MCU
Compiler 说明

目　　录

1. COMPILER 3

2. COMPILER 文件档案限制 4

3. 伪指令 (PSEUDO) 5

4. OBJECT CODE 11

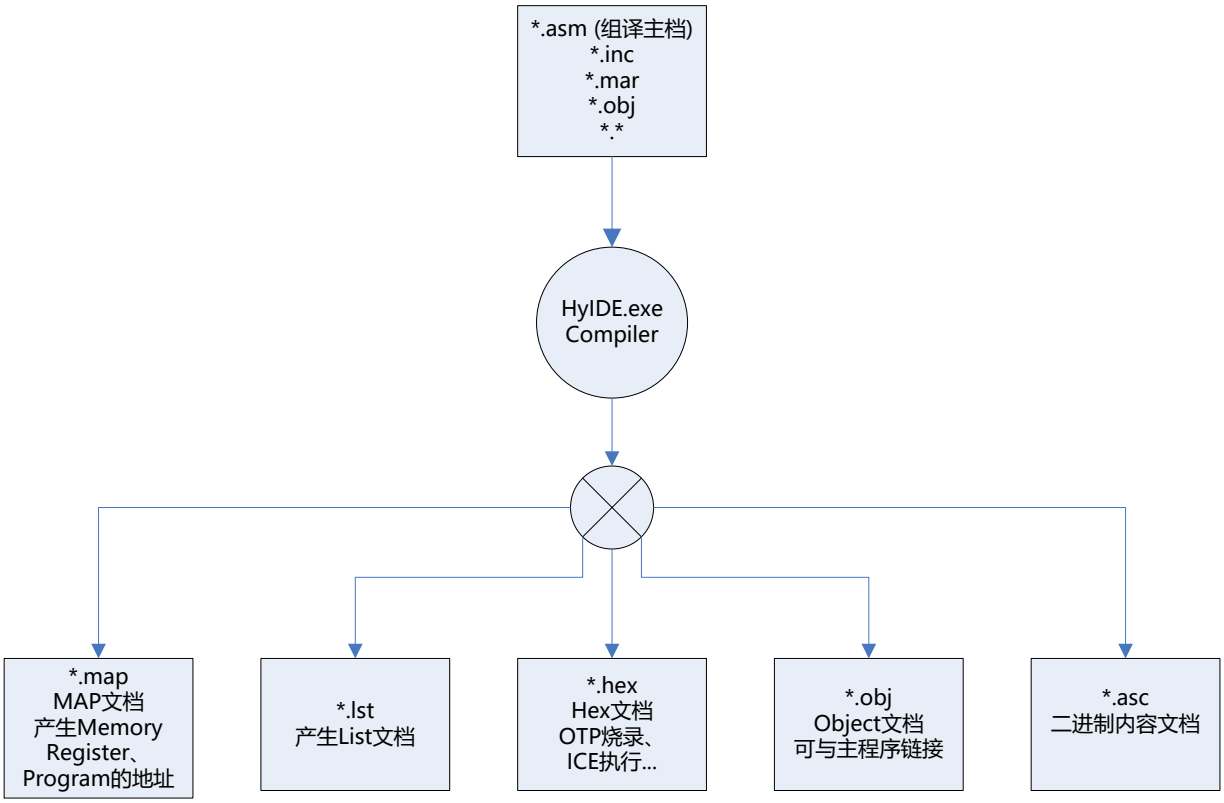
5. 组译 与 组译&&除错的差异 12

6. 错误讯息 12

7. 修订记录 15

1. Compiler

档案产生流程



1.1 组译主档

程序执行主文件，Compiler 主要的文件名，扩展名除 obj、hex、asc、lst、map、msk、pro、err 以及 ifo 外，皆可被 Compiler 接受。

1.2 Object 档

目标链接文件，扩展名为*.obj，可供主程序引用函式 。

1.3 MAP 档

参数的产生文件,扩展名为*.map,参数包括,所有 EQU、DS 定义 Memory 或 Register 的 Address，Object 文件内的 Label 的 Address 。

1.4 List 档

产生 Compiler 后的所有信息，程序的行数，Address，机器码，主程序。

1.5 Hex 档

产生 Hex 文件，供 ICE 程序仿真，OTP 烧录。

1.6 Asc 档

产生机器码的二进制文本文件。

2. Compiler 文件档案限制

1. 文件档案最大为 128k byte， 如果超过最大限制，Compiler 并不会有警告发生，但文件会被毁坏，所以使用者须注意文件档案最大限制。
2. 当文件档案超过 128k 时，需要用 Include 的方式来分开档案，但是每一个文件档案的限制也是 128K，使用者须注意。
3. 文字不论大小写，只要字组相同，皆辨识为相同的文字字组。
4. Include XXXX.obj 档时，必须放置文件最后方。
5. Call .obj 内部的函数时，于该行程序下方，需补上 nopf 函数 这行指令

   例如:

   ```
   CALL  HY17P52WR3
   NOPF      HY17P52WR3
   ```

6. 文件名长度不可超过 128 字符 ，包括文件路劲 Path。

   例如：

   C:\Example\ASM\example.asm ← 此字符串长度不可超过 128 字符
7. 编辑程序每一行的文字长度限制 160 字符。

   例如：

   MVF TEMPABLE , F, BANK ← 此字符串长度不可超过 160 字(包括控格…)
8. Label、Memory 、Register 的定义文字长度最大 32 字符

   例如：

   Label1: ← 此字符串长度不可超过 32 字符

   ECK equ 80h ← ‘ECK’ 不可超过 32 字符
9. Label、Memory 、Register 最多可定义 8192 个(Total)
10. Macro 最大数目为 4096 个
11. Macro 定义的参数个数最大为 32 个。

    例如：

    Exm Macro A,B,C…… ← A,B,C 的参数不可超过 32 个
12. Macro 内引用另外的 Macro 最大层数不可超过 32 层

    例如：

    ```
    ANB   Macro   CC
        Exm
    ENDM
    ANB1   Macro   CC
        ANB
    ENDM
    ANB2    Macro   CC
        ANB1
    ENDM
    ……
    ```

```
ANBn    Macro   CC
    ANBn-1              ← 不可超过 32 层
ENDM
```

3. 伪指令 (Pseudo)

定义程序、内存或特殊寄存器的 Address 时，需用到伪指令来定义，但此指令并不占内存或程序的空间(除了定义程序的伪指令'DB'与'DW'例外)。

Compiler 共有下列伪指令：

INCLUDE	开启另一个文件
ORG	定义程序的 Address
END	程序结束
MACRO	定义模块
ENDM	模块定义结束
EQU	定义立即数
DB	定义一个 Byte 的程序
DW	定义一个 Word 的程序
MEMAR	宣告 SRAM 的 Address
DS	宣告 SRAM 的的长度
EXTENR	引用外部参数
GLOBAL	提供参数给外部程序使用
SET	保留
LOCAL	保留
UP	Label or address / 0x10000
HIGH	Label or address / 0x100
LOW	Label or address mod 0x100
DEFINE	赋值给 IF 语句事件的参数
IF	条件编译语句，判断是否组译 ENDIF 之前的语句
ELSE	否则，配合 IF 的用语
ENDIF	结束 IF 条件编译语句

1. INCLUD

开启主程序内所包括的档案，档案类型可以为 Marco 档案、Object 档案、定义档案以及 Assemble 档案。

如果 INCLUDE 的是 Object 或 Assemble 档案，Compiler 会按照顺序，将指令安排在 ROM 的

Address 內。

例如：

SYSINI.asm 的内容为

```
CLRF    080h
CLRF    081h
.....
```

MAIN.asm 的内容为

```
ORG     0000h
RJ      START
INCLUDE   SYSINI.asm
ORG     0100h
START:
....
```

Compiler **MAIN.asm** 后

```
Address         Program
0000h           RJ        100h
0001h           CLRF      080h
0002h           CLRF      081h
.....
0100h
....
```

如果是 Marco 或定义档案，是不会被安排在 ROM 的 Address 中，除非在程序有呼叫到 Marco，才会将 Marco 内的程序安排在相对应的 ROM Address。

例如：

Def.mar 的内容为

```
            W       EQU       0
            F       EQU       1
            ACCE    EQU       0
            BANK    EQU       1
ADDWORD   MARCO   A , B
            LBSR    B
            MVF     B, W, BANK
            LBSR    A
            ADDF    A, F, BANK
            LBSR    B
            MVF     B, W, BANK
            LBSR    A
            ADDC    A, F, BANK
ENDM
```

MAIN.asm 的内容为

```
INCLUDE     Def.mar
BUFA      EQU     080h
BUFB      EQU     102h
ORG       0000h
GOTO      START
....
ORG       0100h
START:
MVL       012h
MVF       BUFA, F, ACCE
```

```
MVL       034h
MVF       BUFA+1, F, ACCE
MVL       056h
LBSR      BUFB
MVF       BUFB, F, BANK
MVL       078h
MVF       BUFB+1, F, ACCE
ADDWORD   A, B
….
```

Compiler **MAIN.asm** 后

```
Address       Program
0000h         GOTO    START
…..
0100h         MVL     12h
0101h         MVF     80h, 1, 0
0102h         MVL     34h
0103h         MVF     081h, 1, 0
0104h         MVL     56h
0105h         LBSR    1
0106h         MVF     02h, 1, 0
0107h         MVL     78h
0108h         MVF     03h, 1, 0
0109h         LBSR    1
010Ah         MVF     02h, 0, 1
010Bh         LBSR    0
010Ch         ADDF    80h, 1, 1
010Dh         LBSR    1
010Eh         MVF     03h, 0, 1
010Fh         LBSR    0
0110h         ADDC    81h, 1, 1
 ….
```

2. ORG

指定程序内存的地址。

例如：

```
        ORG     0000h
        RJ      Begin
        ….
        ORG     0100h
Begin:
        MVL     10h
        ….
```

Compiler 后

```
0000h     RJ     101h
….
0100h     MVL    10h
```

3. END

程序结束。

当程序中如果有 END 的伪指令，则 Compiler 就会立即中止，不再对 END 以下的程序组译。

4. MACRO

宏定义指令。

可将指令宏定义成一个模块程序，方便程序引用。

例如：

```
MOVFW     MACRO     A, B
MVF    A, 0, B
ENDM

MOVWF     MACRO     A, B
MVF    A, 1, B
ENDM
ORG       0000h
MOVFW     80h,0
MOVWF     10h,1
```

Compiler 后

```
0000h       MVF     80h, 0, 0
0001h       MVF     10h, 1, 1
```

5. ENDM

宏定义指令结束。

宣告 MACRO 后，需有 ENDM 做结束。

6. EQU

定义立即数。

可利用 EQU 来宣告 SRAM 的地址。

例如：

```
COUNT     EQU     80h
```

COUNT 的地址等于 80h

7. DB

定义 1 个 Byte 的程序。

DB 需要双数个参数，如果是单数，则会在前面补上 00h，来合成一个 WORD。

例如：

```
ORG     100h
DB        012h
DB        053h,046h
```

Compiler 后

```
0100h       0012h
0101h       05346h
```

8. DW

定义 1 个 WORD 的程序。

例如：

```
ORG     100h
DW        01234h
```

Compiler 后

```
0100h       01234h
```

9. MEMAR

宣告 SRAM 的 Address。

Compiler 会由 MEMAR 后面的参数来定义 SRAM 的起始地址。

10. DS

宣告 SRAM 的的长度。

例如：

```
MEMAR     080h          ←  SRAM 由 080h 开始编排
TEMP      DS    16      ← 定义 TEMP 有 16 byte (由 080h  到 08Fh)
COUNT     DS    2       ← 定义 COUNT 为 2 个 Byte，地址
090h,091h
BUF       DS    1       ← 定义 BUF 为 1 个 Byte，地址为 092h
```

11. EXTERN

引用外部参数。

在建立 Object Code 时，如果要引用外部程序所定义的参数，不须在本程序中定义，可藉由 EXTERN 的宣告来完成。

例如：

```
        EXTERN    TEMP1, TEMP2
  GLOBAL      ADDBLK

        LEN     EQU     3
ADDBLK:
        MVF     TEMP2, 0, 0
        ADDF    TEMP1, 1, 0
        …….
```

12. GLOBAL

提供参数给外部程序使用。

在建立 Object Code 时，可以将本程序中的参数或子程序释出，提供给外部的程序引用。

13. UP

取数值的最高字节 → 数值 / 0x10000

例如：

```
MVL     UP   Label1   →  UP   Label1 = 0x1000 / 0x10000 = 0x00
….
ORG     01000h
Label1:
```

14. HIGH

取数值的高字节 → 数值 / 0x100

例如：

```
MVL     HIGH   Label1   →  HIGH   Label1 = 0x1000 / 0x100 = 0x10
….
ORG     01000h
Label1:
```

15. LOW

取数值的低字节 → 数值 MOD 0x100

例如：

```
MVL     LOW   Label1   →  LOW   Label1 = 0x10F2 MOD 0x100 = 0xF2
….
ORG     010F2h
```

Label1:

16. DEFINE 、IF、ELSE and ENDIF

DEFINE：赋予数值给 IF 语句事件的参数。

条件语句 IF

用法：

IF 事件

内容 1

ELSE

内容 2

ENDIF

说明:

- DEFINE 的变量与 EQU 定义的变量名称可以相同(两者无关联)
- 当事件成立执行内容 1，否则执行内容 2
- 事件成立可以为"TRUE" OR "1"
- 事件可以为 DEFINE 中的变量
- 可以在 MACRO 内用

限制：**不允许 IF 多层嵌套（如不允许 'ELSE IF 事件' 操作）**

例如：

```
Define      FSR0 = 0
Define      FSR1 = 1
Define      H08B = 0

MOVLF       MACRO   RAMLabel , FSR
            MVL     RAMLabel
        IF  FSR = 0
            MVF     FSR0L, F, ACCE
        ELSE
            MVF     FSR1L, F, ACCE
        ENDIF
ENDM

            ORG     0
            JMP     BEGIN
    ……
    ……
BEGIN:
            MOVLF   TEMP, FSR0
    ……
    ……
    FSR0L   EQU     010h
    FSR0H   EQU     012h
    W       EQU     0
```

```
F               EQU     1
ACCE            EQU     0
BANK            EQU     1
TEMP            EQU     080h
TEMP1           EQU     081h
```

组译后

```
        ORG     0
        JMP     BEGIN
……

……
BEGIN:
        MVL     TEMP
        MVF     FSR0L, F, ACCE
……
……
```

4. Object Code

- 程序代码以二进制形式存放。
- 透过伪指令 Global 产生函式给主程序引用。
- 透过伪指令 Extern 来引用外部的参数或程序
- 主程序利用 Include 的方式来将 Object Code 包含进来。
- 透过 MAP 档案可以知道哪些函式可以被引用呼叫。

```
================================================================================

Memory , Register and Const

================================================================================

Name        Location        Address

                    ----        --------        -------

                  INDF0         Const           0x0000

                 POINC0         Const           0x0001

                 PODEC0         Const           0x0002

================================================================================

Local   Program   Address

================================================================================

Name        Location        Address

                    ----        --------        -------

                 RESET          Program         0x0000

              MAINLOOP          Program         0x000C

         MAINLOOP_DATA          Program         0x0025

             MAINLOOPAA         Program         0x0100

                  TEST          Program         0x0265

================================================================================

Extern   Program   Address
```

```
================================================================================
Name        Location      Address
                          ----        --------      -------
                          CLEAR       Program       0x0102
                          MULTIPLY    Program       0x0106
                          DIVIDE45    Program       0x0128
                          DIVIDE      Program       0x0145

HTOB        Program       0x0186
                          BTOH        Program       0x01B9
                          SQRT        Program       0x01F0
                          TT1         Program       0x025F
                          TEMM        Program       0x025C
```

5. 组译 与 组译&&除错的差异

组译

1. 如果有宣告外部参数(EXTERN)，而在程序段有引用到外部参数，组译皆能成功。
2. 组译不会判断 RAM 或 Program 是否有超出芯片规格范围，因此选择任一芯片皆可组译。

组译&&除错

1. 在程序段有引用到外部参数(EXTERN)，皆无法组译，并显示错误讯息。
2. 组译会检查 RAM 或 Program 是否有超出芯片规格范围，只要超过 Program Size 或引用的 RAM 不存在皆会显示错误讯息。

6. 错误讯息

1. ERROR[Code 0] : Unknown error.
   指令名称错误。
   指令名称错误或是没有在 Macro 中定义名称。
2. ERROR[Code 1]: Couldn't open source file
   找不到 Source 档案。
3. ERROR[Code 2]: Couldn't open file
   Include 文件名错误，或找不到文件名。
4. ERROR[Code 3]: Temp file creation error
   临时文件开启错误。
   重复开启档案。
5. ERROR[Code 4]: Duplicate label or redefining symbol
   重复定义 Symbol 或 Label
6. ERROR[Code 5]: Undefined Symbol

未定义 Symbol 或 Label

7. ERROR[Code 6]: Out of memory...

   Register、Memory or Label 定义超过 8192 个，或 Macro 宣告超过 4096…

8. ERROR[Code 8]: Illegal Symbol

   Register、Memory or Label 定义名称与指令名称相同

9. ERROR[Code 9]: Illegal argument

   LBSR 参数大于 16

10. ERROR[Code 10]: Overwriting previous address contents.

    Address 位置使用重叠

    例如：

```
ORG     0
Nop
Nop
Nop
Nop
JMP     Start
ORG     0004H
```

11. ERROR[Code 11]: Addresses above …..

    程序超过 ROM Size

12. ERROR[Code 13]: Missing argument...

    指令后面的参数不足

    例如：

```
MVF   080h
```

13. ERROR[Code 14]: Too many arguments...

    指令后面太多参数

    例如：

```
CLRF   080h, 1, 0
```

14. ERROR[Code 15]: Undeclared variable

    未声明的变量

15. ERROR[Code 16]: Illegal Macro...

    不合法的 Macro

~~16. ERROR[Code 17]: Illegal Condition~~

~~无效的状态~~

~~17. ERROR[Code 18]: Out of Range...~~

~~超过 Range~~

18. ERROR[Code 19]: Macro nested too deep...

    Macro 内再引用的 Marco 嵌套宏定义超过 32 层

19. ERROR[Code 20]: Circular file reference...

    重复引用档案，或是 INCLUDE 后面没有文件名

20. ERROR[Code 21]: Circular macro reference...

    重复定义 Macor

~~21. ERROR[Code 22]: Use the software illegally~~

~~非法使用软件~~

22. ERROR[Code 23]: Over Range

    超过 RJ 或 RCALL 范围

23. ERROR[Code 24]: illegally Memory or Registor

    定义到不存在的寄存器或内存

24. ERROR[Code 25]: illegally Bit

    定义到不存在的 bit

    例如

    INTF1 - ADCIF - - TMAIF WDTIF E1IF E0IF

    BSF INTF1, 7, 0

25. ERROR[Code 26]: Memory Over

    定义超出内存范围

7. 修订记录

以下描述本文件差异较大的地方，而标点符号与字形的改变不在此描述范围。

版本	页次	变更摘要
V01	All	新增
V02	04	新增 Include .obj