# 1:头文件扫描步骤：

1：从当前文件夹（一般当前文件就是debug目录下）找到所有的头文件，或者直接给定工程目录的路径，遍历子文件夹，从中找到所有后缀为.h的文件。根据经验一些头文件可以去掉，节省一点时间

2：读取每个头文件，读取其中的每一行，然后判断是否读到文件末尾，没到末尾则解析该行，判断该行的内容属于哪个分类。

解析步骤：

1：是否是空行，直接返回

2：去掉该行开头的空格。

3：找到//的位置，假如在开头，那么整行都是注释，直接可以返回。

4：如果//不在开头并且没有找到//，那么就去掉后面的空格。如果不在开头并且找到了//，那么就先去掉注释，然后去掉后面的空格。

5：找关键字，关键字有"typedef","struct","union","enum"，关键字要求都是typedef与其它关键字的组合，没有组合关键字的行都可以忽略。

6：根据找到的关键字可以接着获取对应的信息，例如如果是结构体，那么就需要找到“{”，然后看是否读取到“}”， 如果没有，如果是空行直接返回，否则先去掉开头的空格，然后找到//的位置，假如在开头，那么整行都是注释，直接可以返回。如果不在开头，那么搜索第一个“;”的位置，如果没有分号，那么说明忘记写分号了，如果有就获取子字符串，然后去掉两边的空格。然后通过空格分割字符串得到类型和名称，如果是bit的还有bit的位数。假如是数组的还需要有特殊处理。搜索完成后就将该结构体放入一个map中（或者对应结构中）。联合体和枚举类型一样。

3：根据名称去掉一些不在树状结构中的结构体。这一步也可以通过第一步的去掉某些头文件来实现。

4：构建树状结构。首先将最顶层的父元素作为根（需要手动输入其对应属性）。然后嵌套调用函数。

# 2：头文件格式的几点要求：

1：所有结构体，枚举，联合体都使用typedef定义

2：位操作的：和数字之间要求有空格。

3：不扫描DspCpu.h. FunLutTbl.h。VirFloatMath.h。ServoApp.h

4：”(\*” “);” 这两个字符串要写到一起，中间不能有空格，定义函数指针的时候

5：sev\_obj[SERVO\_AXIS\_NUM] 数组之间不能有空格

6：结构体，联合体，枚举 结束时“}name;“不能有空格。

7：不认为是数组

SERVO\_SCHD sev\_chd[SERVO\_AXIS\_NUM];

SERVO\_TASK sev\_task[SERVO\_AXIS\_NUM];

SERVO\_OBJ sev\_obj[SERVO\_AXIS\_NUM];

# 3：在后台实现时的方式

该功能要集成到调试软件后台中，因为没有办法一步完成，因为中间必然有一个编译的过程，编译可以在vs做或者在IAR中做，

1：在vs做需要将代码拷贝到vs的工程下，然后添加到vs工程中，定义gSevDrv然后编译，这样就没法在后台中实现了，因为需要源码级别的操作。并且这种方式不能保证偏移地址完全正确（因为编译器不同）。

2：在IAR下做，首先需要在vs下生成一个文件，放到IAR工程中，然后编译获取地址，将地址写入一个txt文本，然后再拿回到VS中读取。最终生成树状结构。这种方式可以保证偏移地址完全正确。并且在后台中有可行性。