IAR EWARM 链接时保留未引用的段

IAR 的 Linker 在执行链接时，为了减小生成代码的大小，如果 library 或可重定位目标文件中的某些函数或者变量没有被引用，这些函数或变量会被丢弃，并不会被链接到可执行文件中。但有的时候，用户出于某些原因，需要保留未引用的函数或者变量，IAR 也提供了相应的方法。

1. C/C++源码中可以使用“*__root*”关键字，在定义时强制保留函数或变量。例如，要保留程序中未调用的 GenerateRandomNumber()函数。

使用“*__root*”关键字：

__root void GenerateRandomNumber ()

2. 但是如果要保留的是 library 中的函数，就没法使用“*__root*”关键字了。此时，可以使用 Linker 选项“--keep symbol”指令强制链接程序中未使用的符号，它作用范围是整个工程，不论需要保留的符号是在源码文件中还是 library 中都适用。

在 Linker 选项中使用“--keep”命令，保留 GenerateRandomNumber ()函数：

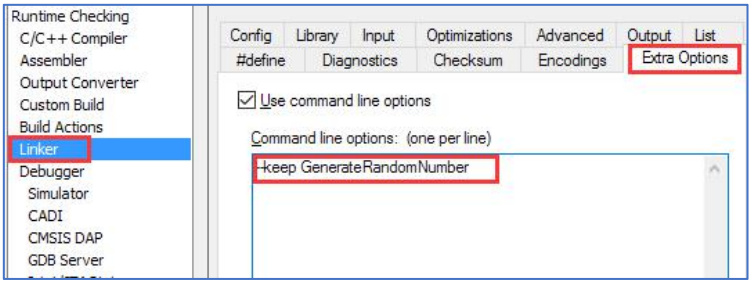

图 1

3. “--keep”命令适用于需要保留的符号为数不多的情况，如果有大量的符号要强制保留，就不合适了。这种情况可以使用“--no_remove”或“--no_fragments”命令。

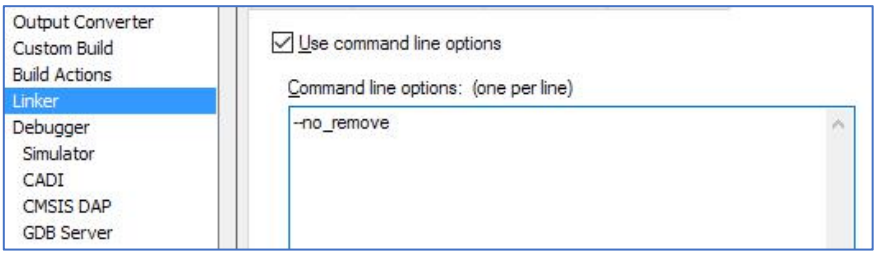

图 2

“--no_remove”和“--no_fragments”的作用范围也是整个工程。

除了上述的三种情况之外，如果只想保留某几个 library 或目标文件中的某几个段，其他的段不需要进行链接，以达到保留所需要的段的同时，尽量减少代码的尺寸，可以在 ILink 配置文件中使用 keep 命令。

icf 文件 keep 命令格式如下：

keep { section-selectors }[except { section-selectors}];

section-selectors 是 section 的选择格式：

[section-attribute] [section section-name][object {module|filename}]

section-attribute：

ro [code|data] | rw [code|data] | zi

只有指定了下列属性的 section 才会被选择：

ro|readonly, for read-only sections.

rw|readwrite, for read/write sections.

zi|zeroinit, for zero-initialized sections.

section section-name：

section 的名称，只有匹配名称 section 才会选中，有 2 个通配符可以使用：“?”和“*”。

?：匹配任何一个字符

*：匹配 0 或者多个字符

object {module|filename}:

匹配静态库中的模块或者目标文件中的 section 才会选中。

except{}是可选项，用于排除要不需保留的段。

示例：保留 Lib_section.a 库中除了.function2 以外属性为只读的段。

(1). 从 IAR EWARM 的安装目录下将对应的链接器配置文件复制到工程目录中，避免对原始的 icf 配置文件作修改。然后进入工程的 Options 选项，选择 Linker -> config，勾选 “Override defualt”，单击旁边的按钮，选择之前复制到工程目录的 icf 文件。推荐采用$PROJ_DIR$\相对路径，如图 3 中的样式。

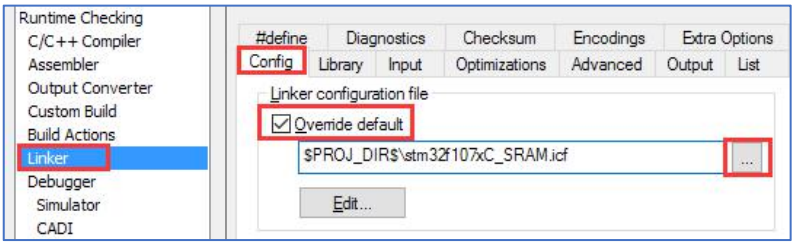

图 3

(2). 用文本编辑器打开 icf 文件，可以在后面加上 keep 指令：

```
36  place at address mem:__ICFEDIT_intvec_start__ { readonly section .intvec };
37
38  place in ROM1_region   { section .rodata,
39                           section .myconst }; /*constants*/
40  place in ROM2_region   { section .text, section .mycode,
41                           section .data_init,
42                           section .iar.init_table,
43                           block FUNCBLOCK};
44
45  place in RAM1_region   { section .mydata};
46  place in RAM2_region   { readwrite,
47                           block CSTACK,
48                           block HEAP };
49
50  keep {readonly object Lib_section.a} except {section .function2 };
51
```

图 4

此后，构建程序时，Lib_section.a 中除了.function2 以外的只读的段，不论是否在程序中被引用，都会保留，链接结果可用通过 map 映射文件来确认。