Importance of Map File while Debugging Memory related Bugs

Amrita Pandey
Embedded C, C++, Python
Published Jan 2, 2021

+ Follow

We all will agree that in order to build a correct product, validation techniques play a very vital role throughout the product lifecycle of an embedded system. Finding bugs in the initial stages of development process is surely something that each Embedded Engineer strive for.

Understanding the memory allocation in embedded system is very essential. It can help in debugging memory related bugs. One should have a basic knowledge of different sections of memory namely Text, Data, Heap and Stack, in order to read and interpret map file.

Below Table briefly describes various memory sections:

	Section	Description
1	Text(.section .text)	Contains Code(instructions)
2	Read-Only Data(.section .rodata)	Contains pre-initialized constants
3	Read-Write Data(.section .data)	Contains pre-initialized variables
4	BSS(.section .bss)	Contains un-initialized data
5	Heap	Dynamic memory allocation takes place in heap
6	Stack	Initialized and uninitialized automatic variables

What is a map file?

A map file is generated by the linker and the format of the file will be different for each linker. It tells where in memory variables and functions are located. So, a map file is simply a table of symbols, their locations and their sizes. Usually there will be an overall summary of memory usage for static data and code space. Some linkers can place other information in a map file, such as stack usage analysis for each function. The critical values being those for main() and any task/thread and ISR entry points. Some may also generate cross-reference tables or call graphs. So, in order to generate relevant information in map file, the 'Linker Script' needs to be written accordingly.

Generating the Map File

If you are using any particular IDE, then you can enable the map file generation options in IDE else, the command for generating the map file as part of your build, depends on the compiler which you are using. You can easily find the linking option for generating map files by looking into Linker documents provided by the respective compilers.

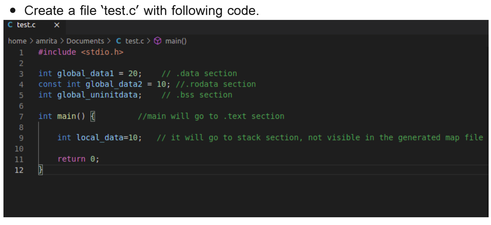
For example if you are using GNU, then the option for generation is:

```
-Wl,option
```

Pass *option* as an option to the linker. If *option* contains commas, it is split into multiple options at the commas. You can use this syntax to pass an argument to the option. For example, -Wl,-Map,output.map passes -Map output.map to the linker. When using the GNU linker, you can also get the same effect with -Wl,-Map=output.map

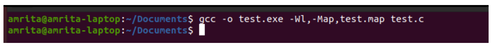
Map file demonstration using simple c code-

- Create a file 'test.c' with following code.



- To compile the file 'test.c' and to generate map file with gcc, use the following command:

```
$ gcc -o test.exe -Wl,-Map,test.map test.c
```



By using above command, executable file 'test' and map file 'test.map' would be created. You can check for the declared variables in the map file. The map file will clearly show the section in which variable is present, address of variable and size of the variable.

- Below are the screenshots from 'test.map' file:

Hence, map file provides valuable information that can help you understand and optimize memory. They can be read anywhere and anytime, without requiring a supporting program and without requiring all your program's binaries to get the same information. I highly recommend keeping that file for any firmware running in production.

#Mapfile #Linker #MemoryLayout #Memory #Embedded #Software #Embeddedtesting #MapfileInterpretation

To view or add a comment, sign in

Sign in
Stay updated on your professional world
Sign in
Continue with Google
New to LinkedIn? Join now

Others also viewed
C Build Process in details
Abdelaziz Moustafa · 3y
How does Linker Works???
Vinit Tirnagarwar · 1y
Differences between static and dynamic libraries
Juan David Tuta Botero · 2y
Startup code
Ahmed Abd El-Ghafar Mohammed · 3y
The Functional Safety Mirror
Hassan Higazy · 4y
The Functional Safety Mirror
Hassan Higazy · 4y
Show more
Explore topics
Sales
Marketing
Public Administration
Business Administration
HR Management
Engineering
Soft Skills
See All

LinkedIn © 2023 About Accessibility User Agreement Privacy Policy Your California Privacy Choices Cookie Policy Copyright Policy Brand Policy Guest Controls
Community Guidelines Language