Lib Files 101

Vineel Kovvuri
http://vineelkovvuri.com

1 Introduction

During the compilation one of the crucial step after assembling is creating the Object files. The collection of these object files is called a lib file. We can make these .lib files when creating below two type of projects 1. Static Library 2. Dynamic Linked Library

The format of these .lib files is specified in 'Archive (Library) File Format.' section of PE Format specification. As per the spec, .lib is an archive of individual .obj files with some metadata. Multiple tools can be used to extract lib files. Visual Studio installation contains Lib.exe tool. Because .lib and .obj files follow Unix COFF format Unix binutil's 'ar' tool can be used to extract it.

```
Lib StaticLib1.lib /list
Lib StaticLib1.lib /EXTRACT:Debug\sub.obj /out:sub.obj <-- Extracts one file at a
↪  time!

or

ar -x StaticLib1.lib

or

use 7-Zip to extract it
```

2 Static Library

Static Library is created when you want to provide the complete code to link into another dll or exe. For example, If a static library project contains 4 files add.c,sub.c,mul.c,div.c containing functions for their operations like below respectively.

```
int add(int a)
{
    return a + 1;
}
```

When you build the project what we get is a .lib file containing obj files of each of the above .c file. So a static library contains all the code that gets compiled from your project, and this .lib will be consumed by any other project types(dll or exe).

NOTE: The functions inside a Static Library is not declared with __declspec(dllexport) because all functions declared in a static library are meant to be consumed/included by others directly.

```
dumpbin /exports StaticLib.lib  <-- shows nothing because .lib itself does not
↪  export anything
dumpbin /symbols StaticLib.lib  <-- shows all the symbols present
```
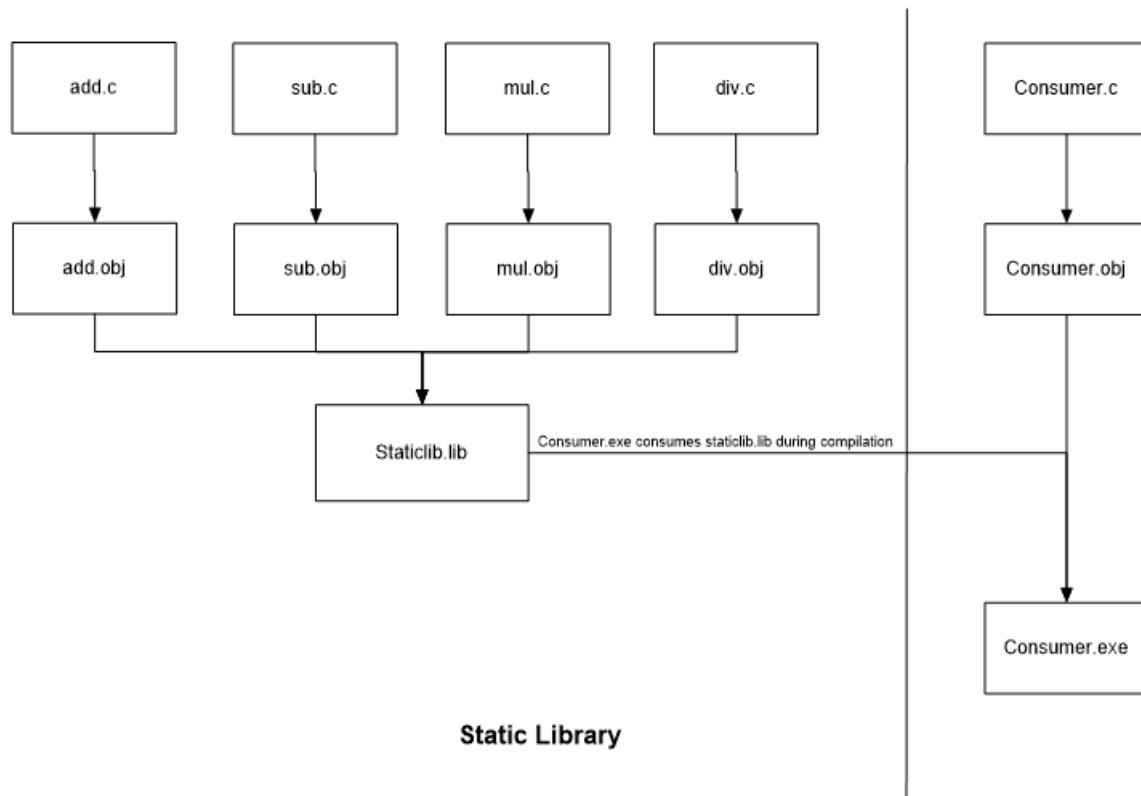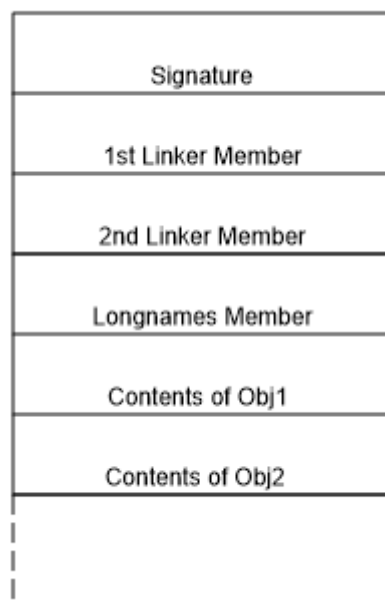
Figure 1: Static Library

Figure 2: Concluded based on HxD view of the file

3   Dynamic Linked Library

Dynamic Linked Library(DLL) is in many ways similar to Static Library because it also provides the code to be used by other projects like dll or exe, but the difference is in the way the code gets used by them. In DLL, the outcome of the project is not only a .lib file but also a .dll file. In fact, in case of DLL project, the .lib file does not contain any .obj file instead it contains only pointers to which dll

includes the code and how other projects can refer to it in their compilation. In Dll, all of the code is indeed present inside the .dll file.

Hence this is also the reason why we can only see the exported symbols but not their code in case of dll's .lib file.

```
__declspec(dllexport) int add(int a)
{
    return a + 1;
}
```

```
dumpbin /exports Dll1.lib <-- shows all the exported functions
```
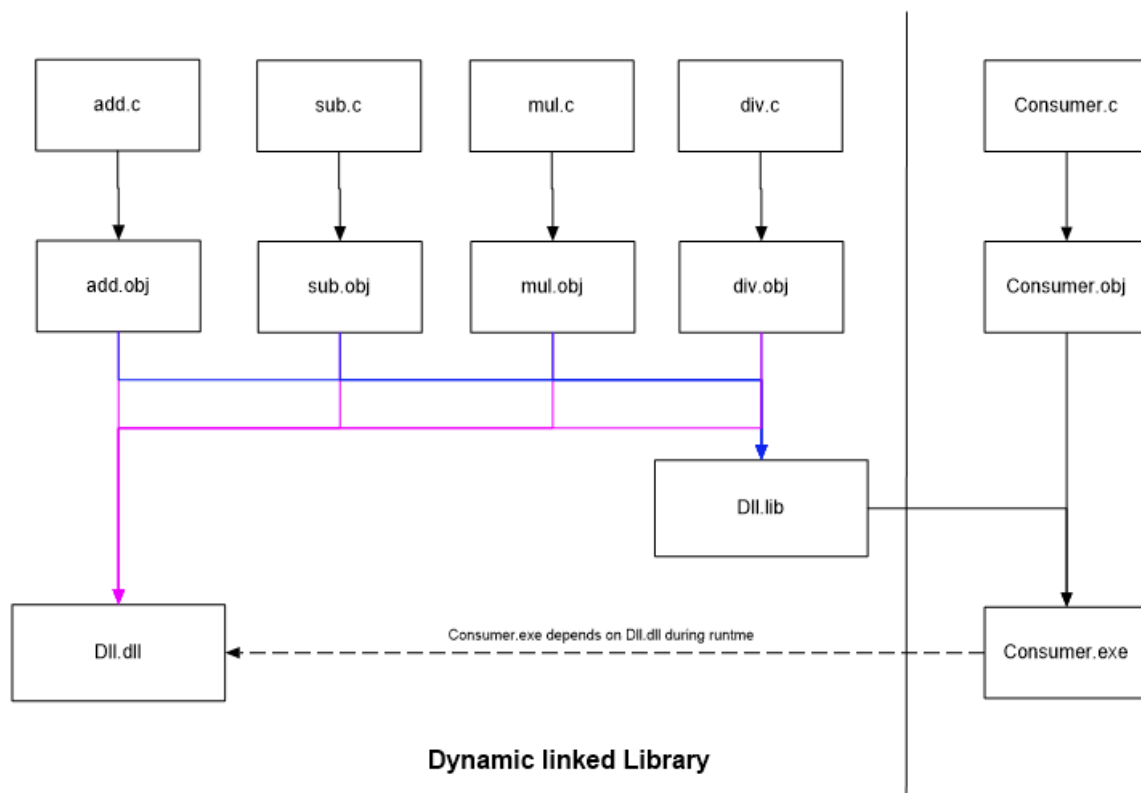

Figure 3: Dynamic Linked Library

Figure 4: Concluded based on HxD view of the file

NOTE: The functions inside a Dynamic Linked Library need to be declared with __declspec(dllexport) if they need to be visible and consumed by others(indirectly).

4   References

1. PE Format

4