基础5-链接与链接脚本

1.链接器的行为准则

在程序链接时，链接器只关心函数和全局变量，链接器把它们识别为符号，来进行链接。注意，高级语言中的函数重载仅仅是语法糖，本质是不同的函数

强符号：函数和已初始化的全局变量
弱符号：未初始化的全局变量

- 不允许有多个强符号。如下，编译器会报错

```
//a.c中定义的全局变量i
int i = 10;

//b.c中定义的全局变量i
double i = 10;
```

- 若有1个强符号和多个弱符号，则选择强符号。如下，不会报错

```
//a.c中定义的全局变量i
int i = 10;

//b.c中定义的全局变量i
double i;
```

- 若有多个弱符号，随机从它们里面选一个。如下，不会报错

```
//a.c中定义的全局变量i
int i;

//b.c中定义的全局变量
double i;
```

- 由此可看出，滥用全局变量很危险，所以应该尽量避免全局变量，或使用static修饰

2.链接脚本分析

以u-boot.lds为例，位于根文件夹下/board/samsung/x210内，它是U-boot的总链接脚本。

- 本段最开始指定了输出的格式，然后指定输出的架构为arm架构
- 指定整个程序的入口地址，可以认为是第一句指令，_start是start.S的第一个lable
- 值得注意的是，程序入口并不代表它位于存储介质的起始位置。一般起始位置存放的是16字节校验头和异常向量表

```
OUTPUT_FORMAT("elf32-littlearm", "elf32-littlearm", "elf32-littlearm")
/*OUTPUT_FORMAT("elf32-arm", "elf32-arm", "elf32-arm")*//*这句是注释*/
OUTPUT_ARCH(arm)
ENTRY(_start)
```

- SECTIONS表示正式开始地址划分
- .的意思是当前地址，这句将当前地址（代码段起始地址）设为0x00000000，但是其实这个地址会被config.mk用-Ttext $(TEXT_BASE)指定的虚拟地址0xc3e00000（由顶层Makefile填充给config.mk）覆盖掉

```
SECTIONS
{
    . = 0x00000000;

    . = ALIGN(4);
    .text      :
```

- .text表示开始代码段的链接
- 代码段的链接顺序很重要，首先start.o必须在第一个
- 由于uboot需要重定位，故所有和重定位有关的代码必须链接在最前面，作为16kb的bl1。而其他所有的.o文件就往后任意链接了

```
    .text      :
    {
      cpu/s5pc11x/start.o   (.text)
      cpu/s5pc11x/s5pc110/cpu_init.o    (.text)
      board/samsung/x210/lowlevel_init.o    (.text)
          cpu/s5pc11x/onenand_cp.o      (.text)
          cpu/s5pc11x/nand_cp.o (.text)
          cpu/s5pc11x/movi.o (.text)
          common/secure_boot.o (.text)
      common/ace_sha1.o (.text)
      cpu/s5pc11x/pmic.o (.text)
      *(.text)
    }
```

- . = ALIGN(4)的意思是将当前地址（代码段结束地址）四字节对齐，然后将其作为只读数据段的起始地址（存放只读的全局变量）
- 同理，对数据段（存放全局变量）和got段进行相同设置

```
    . = ALIGN(4);
    .rodata : { *(.rodata) }

    . = ALIGN(4);
    .data : { *(.data) }

    . = ALIGN(4);
    .got : { *(.got) }
```

- 设置自定义段u_boot_cmd，里面存放着的是一个个命令结构体（结构体内都是命令的信息），它们是紧挨着的，其实有点像结构体数组，只不过是乱序的。写出 __u_boot_cmd_start， __u_boot_cmd_endt的地址是为了要在源码中引用这两个地址，由此来使用命令结构体
- 然后设置mmudata段
- 最后设置bss段（存放初始值为0的全局变量），写出 __bss_start，_end就是为了要在.s或.c中引用这两个地址

```
    __u_boot_cmd_start = .;
    .u_boot_cmd : { *(.u_boot_cmd) }
    __u_boot_cmd_end = .;

    . = ALIGN(4);
    .mmudata : { *(.mmudata) }

    . = ALIGN(4);
    __bss_start = .;
    .bss : { *(.bss) }
    _end = .;
}
```