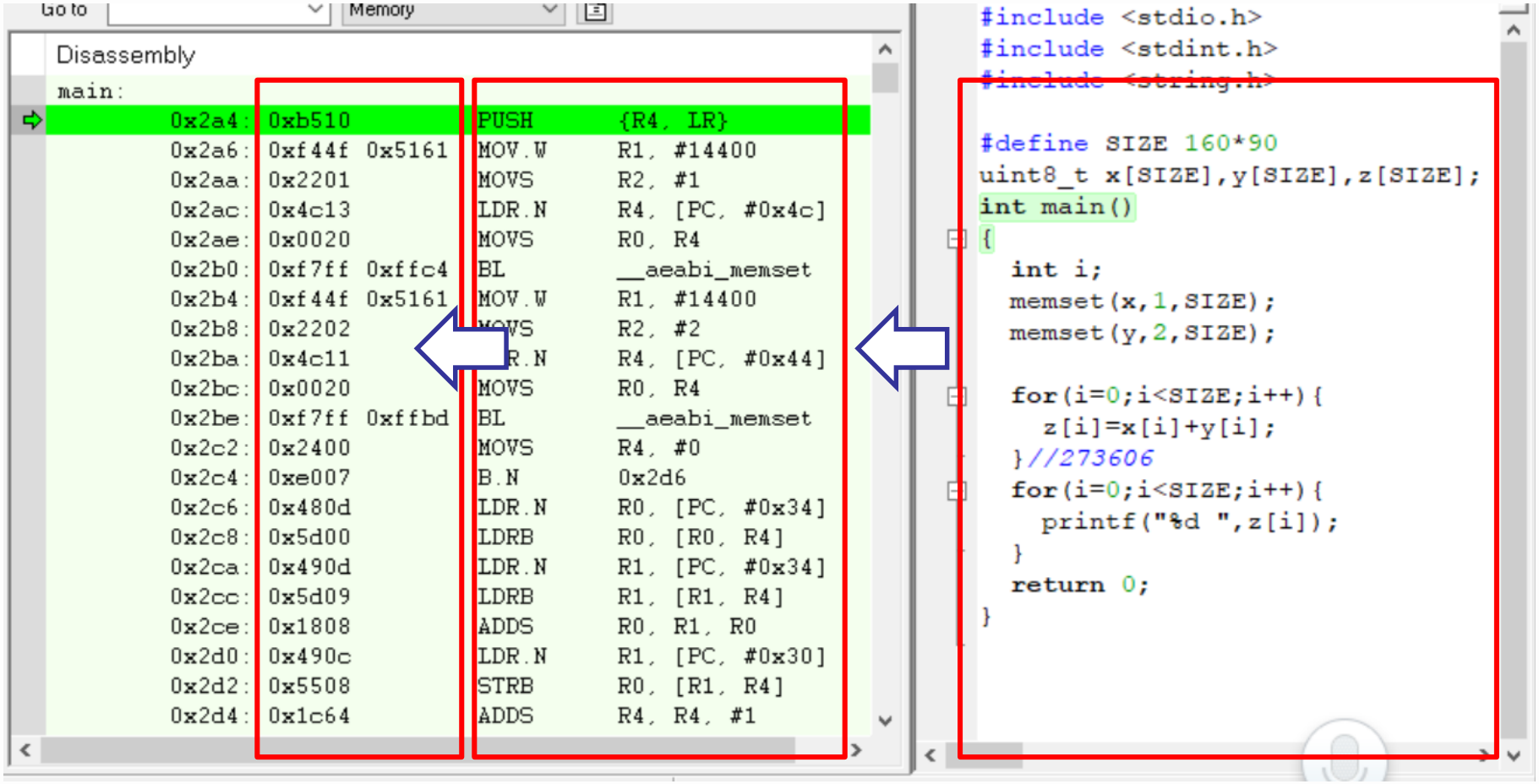
北京交通大学
BEIJING JIAOTONG UNIVERSITY

嵌入式系统

-软件开发环境基础

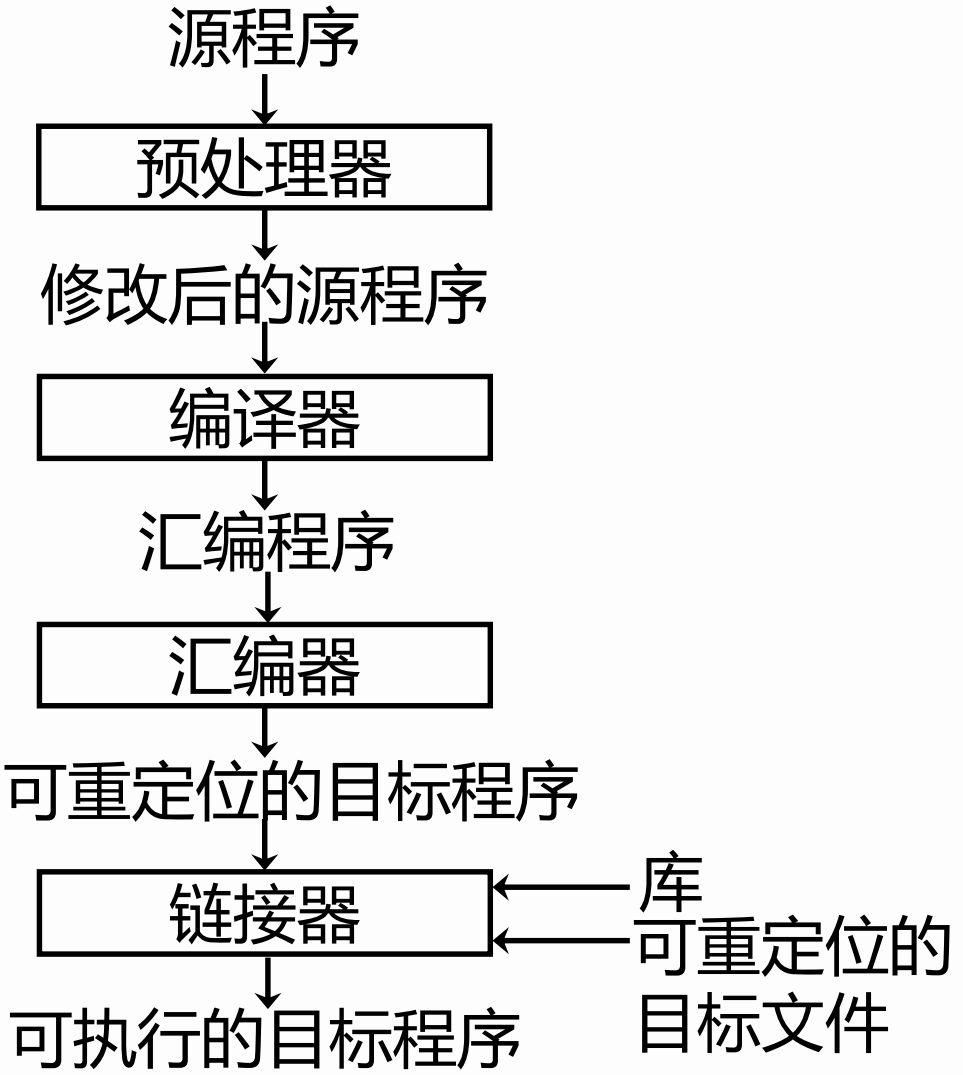
赵翔

xiangzh@bjtu.edu.cn


Go to
Memory
Disassembly
main:
0x2a4: 0xb510 PUSH {R4, LR}
0x2a6: 0xf44f 0x5161 MOV.W R1, #14400
0x2aa: 0x2201 MOVS R2, #1
0x2ac: 0x4c13 LDR.N R4, [PC, #0x4c]
0x2ae: 0x0020 MOVS R0, R4
0x2b0: 0xf7ff 0xffc4 BL __aeabi_memset
0x2b4: 0xf44f 0x5161 MOV.W R1, #14400
0x2b8: 0x2202 MOVS R2, #2
0x2ba: 0x4c11 LDR.N R4, [PC, #0x44]
0x2bc: 0x0020 MOVS R0, R4
0x2be: 0xf7ff 0xffbd BL __aeabi_memset
0x2c2: 0x2400 MOVS R4, #0
0x2c4: 0xe007 B.N 0x2d6
0x2c6: 0x480d LDR.N R0, [PC, #0x34]
0x2c8: 0x5d00 LDRB R0, [R0, R4]
0x2ca: 0x490d LDR.N R1, [PC, #0x34]
0x2cc: 0x5d09 LDRB R1, [R1, R4]
0x2ce: 0x1808 ADDS R0, R1, R0
0x2d0: 0x490c LDR.N R1, [PC, #0x30]
0x2d2: 0x5508 STRB R0, [R1, R4]
0x2d4: 0x1c64 ADDS R4, R4, #1
#include <stdio.h>
#include <stdint.h>
#include <string.h>
#define SIZE 160*90
uint8_t x[SIZE],y[SIZE],z[SIZE];
int main()
{
int i;
memset(x,1,SIZE);
memset(y,2,SIZE);
for(i=0;i<SIZE;i++){
z[i]=x[i]+y[i];
}//273606
for(i=0;i<SIZE;i++){
printf("%d ",z[i]);
}
return 0;
}

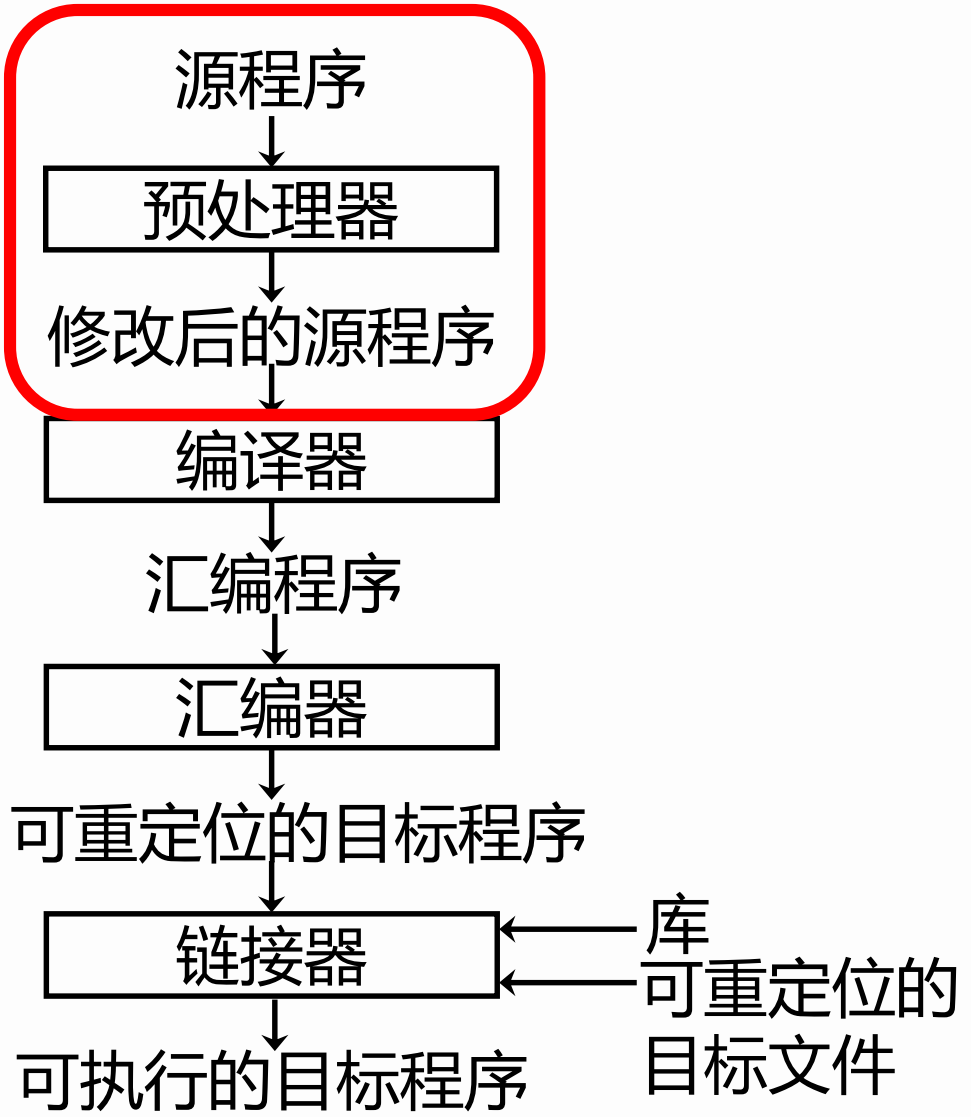
嵌入式软件生成过程

- C源程序可以分成若干个模块(.c/.h文件)
- 分别进行预处理、编译和汇编、形成可重定位的目标文件
- 目标文件和必要的库文件连接成一个可执行的目标文件



嵌入式软件生成过程

- 预处理
  - 处理所有“#”开头的命令
    - #define （替换）
    - #include （展开）
    - #if/#else/#ifdef…
  - 输出处理后的C程序



嵌入式软件生成过程

- 编译
  - 将C语句转换成汇编语句
    - 每个C源文件单独进行
    - 词法分析
    - 语义分析
    - 建立符号表
    - 优化
  - 输出汇编程序（文本）

嵌入式软件生成过程

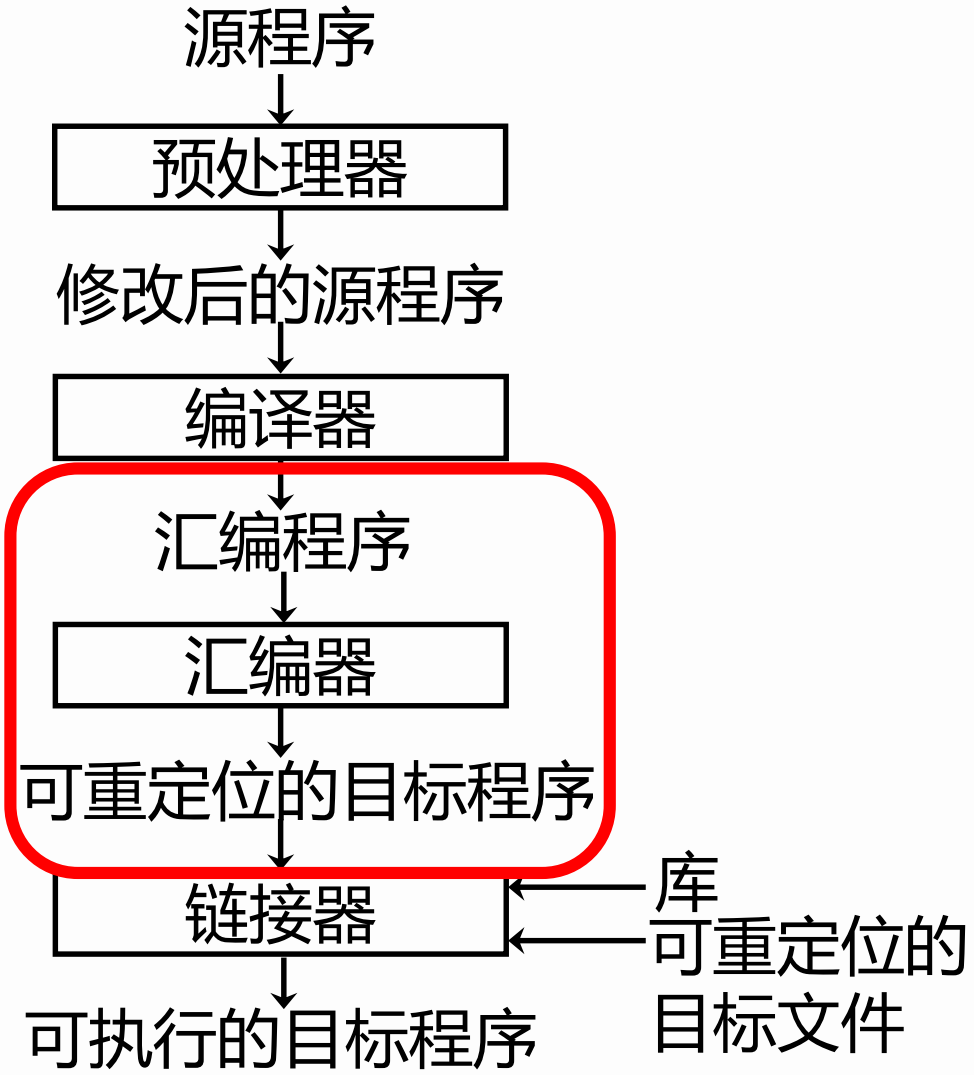
```c
#include<stdio.h>
#include<stdlib.h>
static int var = 0;
int main()
{
  int i,var3=0x5555;
  static int var2 = 0;
  int *buf;
  buf=malloc(100);
  printf("%x,%x,%x,%x\n",buf,&var3,&var,&var2);
  for(i=0;i<100;i++)
  {
        var3+=i;
        buf[i]=i;
  }
  printf("%d",var3);
  printf("%d",buf[25]);
  while(1);
}
```

```
        THUMB
//    4 int main()
//    5 {
main:
        PUSH     {R2-R4,LR}
          CFI R14 Frame(CFA, -4)
          CFI R4 Frame(CFA, -8)
          CFI CFA R13+16
//    6   int i,var3=0x5555;
        MOVW     R0,#+21845
        STR      R0,[SP, #+4]
//    7   static int var2 = 0;
//    8   int *buf;
//    9   buf=malloc(100);
        MOVS     R0,#+100
          CFI FunCall malloc
        BL       malloc
        MOVS     R4,R0
//   10   printf("%x,%x,%x,%x\n",buf,&var3,&var,&var2);
        LDR.N    R0,??main_0+0x4
        STR      R0,[SP, #+0]
        LDR.N    R3,??main_0+0x8
        ADD      R2,SP,#+4
        MOVS     R1,R4
        LDR.N    R0,??main_0+0xC
          CFI FunCall printf
        BL       printf
//   11   for(i=0;i<100;i++)
        MOVS     R0,#+0
        B.N      ??main_1
//   12   {
//   13         var3+=i;
??main_2:
        LDR      R1,[SP, #+4]
        ADDS     R1,R0,R1
        STR      R1,[SP, #+4]
//   14         buf[i]=i;
        STR      R0,[R4, R0, LSL #+2]
//   15   }
```

嵌入式软件生成过程

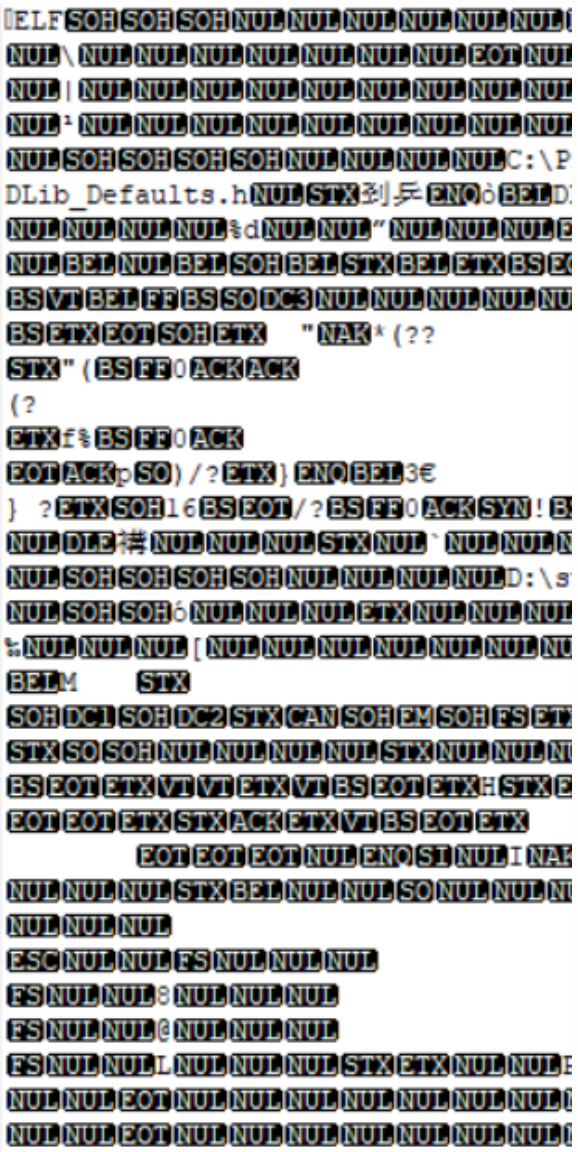
汇编

- 将汇编语句翻译成机器指令（二进制代码）
  - 每个源文件单独进行
- 输出“可重定位目标程序”（二进制）
  - “.o”文件



嵌入式软件生成过程

```
          CFI Block cfiBlock0 Using cfiCommon0
          CFI Function main
        THUMB
//      4 int main()
//      5 {
main:
        PUSH      {R2-R4,LR}
          CFI R14 Frame(CFA, -4)
          CFI R4 Frame(CFA, -8)
          CFI CFA R13+16
//      6   int i,var3=0x5555;
        MOVW      R0,#+21845
        STR       R0,[SP, #+4]
//      7   static int var2 = 0;
//      8   int *buf;
//      9   buf=malloc(100);
        MOVS      R0,#+100
          CFI FunCall malloc
        BL        malloc
        MOVS      R4,R0
//     10   printf("%x,%x,%x,%x|n",buf,&var3,&var,&var2);
        LDR.N     R0,??main_0+0x4
        STR       R0,[SP, #+0]
        LDR.N     R3,??main_0+0x8
        ADD       R2,SP,#+4
        MOVS      R1,R4
        LDR.N     R0,??main_0+0xC
          CFI FunCall printf
        BL        printf
//     11   for(i=0;i<100;i++)
        MOVS      R0,#+0
        B.N       ??main_1
//     12   {
```



嵌入式软件生成过程

目标文件的格式

- 目标文件格式随系统不同而不同
- Unix的ELF（Executable and Linkable Format）格式
- Linux、System V Unix的后期版本、BSD Unix变体和Sun Solaris，都使用Unix的ELF格式

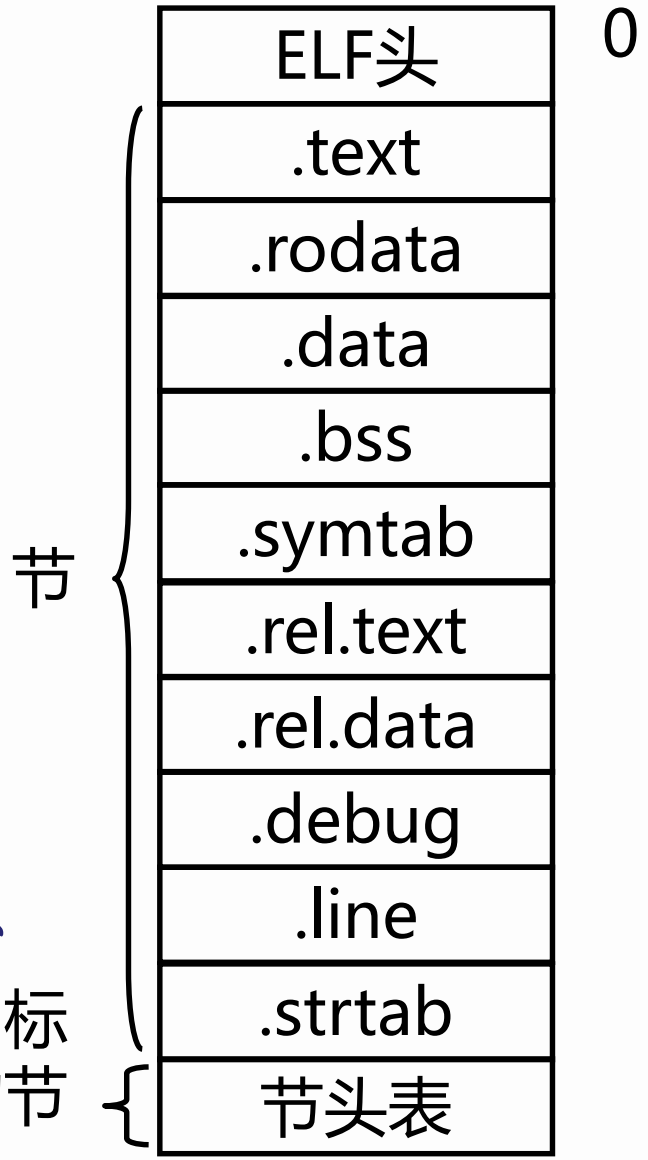
嵌入式软件生成过程

ELF头

- 描述了字的大小
- 产生此文件的系统的字节次序
- 目标文件的类型
- 机器类型
- 节头表的位置、条目多少
- 其它

节头表

- 描述目标文件中各节的位置和大小
- 处于目标文件的末尾



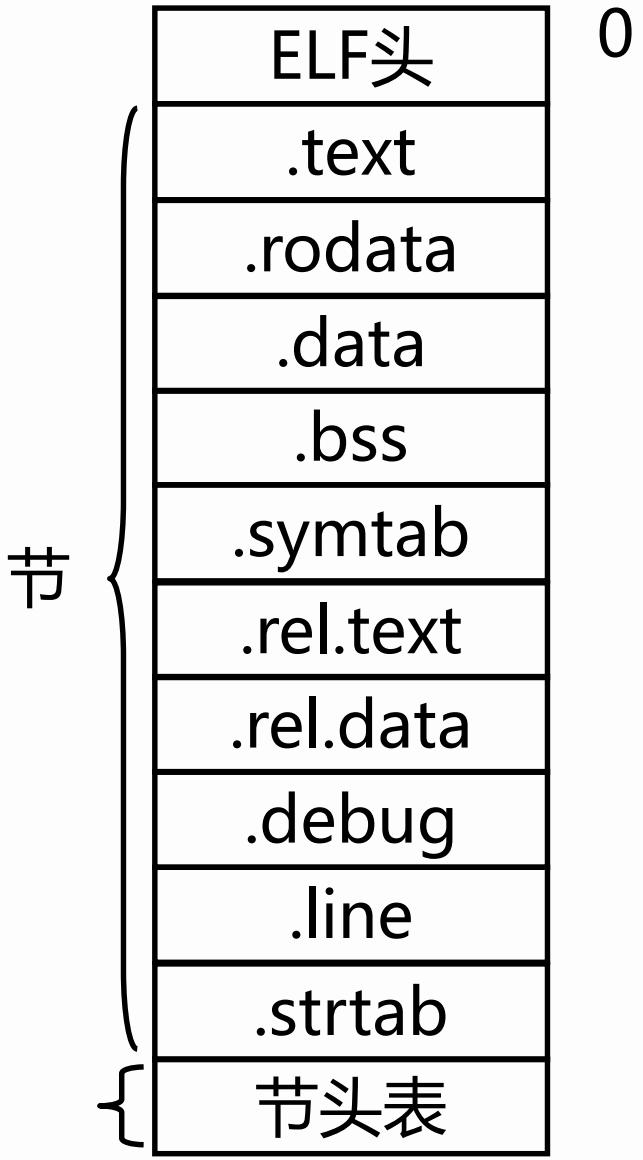
嵌入式软件生成过程

.text节
- 被编译程序的机器代码

.rodata节
- 如printf语句中的格式串和switch语句的跳转表等只读数据

.data节
- 已初始化的全局变量

.bss节（.comm 节）
- 未初始化的全局变量
- 在目标文件中不占实际的空间



嵌入式软件生成过程

.symtab节

- 记录在该模块中定义和引用的函数和全局变量的信息的符号表
  - Name：指针
  - Value：偏移地址或绝对地址
  - Size：字节数
  - Type：FUNC 或 OBJECT
  - Bind：GLOBAL 或 LOCAL 或 EXTERN