四、GUN ARM汇编基础

1.GNU ARM汇编器

GNU ARM 汇编语言基本格式:

- label:instruction or directive or pseudo-instruction @comment
- instruction:机器指令，处理器中有特定硬件来执行;
- direvtive:伪操作，没有对应机器指令，只起编译器指示作用;
- pseudo-instruction:伪指令,会被编译为一条或者多条机器指令;

2.GNU编译环境构成

GNU编译工具主要包括汇编器as、编译器gcc、链接器ld、反汇编工具objdump等，没种工具都有X86版本和ARM版本。

3.段以及LDS文件

GNU ARM以段为单位来组织程序,段是具有相同属性的一段内容,汇编所产生的目标文件至少具有text(存放MOV、SUBS等)、data、bss这三个段,分别对应可执行代码，初始化数据以及未初始化数据。

一般通过LDS链接脚本文件完成段定义的过程，如下面的uart.lds文件:

```
SECTION  // 段声明
{
  .=0x23E00000 //定义段首链接地址
.text:
      {
        start.o //指明首先链接start.o文件
        *(.text)
      }

.bss:
```

```
        {
          *(.bss) //定义bss段并且跟在text段之后
        }

.data:
        {

          *(data) //定义data段
        }

}
```

4.GNU ARM常用伪操作

符号定义伪操作

- .equ symbobl,expr //将symbol定义位expr
- .global symbol //将symbol定义为全局标号
- .extern symbol //声明symbol为一个外部变量

数据定义伪操作

- .word expr {,expr}... //分配子内存单元并用expr初始化
- .byte expr {,expr}... //分配字节内存单元并用expr初始化
- .long expr {,expr}... //同.word
- .ascii expr {,expr}... //字符串，非零结束符
- .string expr {,expr}... //字符串，零结束符同.asciz
- .zreo expr {,expr}... //用0田中size字节的内容

汇编与反汇编控制伪操作

- .arm //定义以下代码用arm指令集编译，同.code32
- .thumb //定义以下代码用thumb指令集编译，同.code16
- .end //汇编结束标志

预定义控制操作

- .if // 条件判断语句
- .else //条件判断语句
- .endif // 条件判断语句
- .macro // 宏定义开始
- .endm //宏定义结束
- .exitm // 中途跳转出宏
- .include "file_name" //包含文件标识

```
/*---宏定义以及使用实例---*/
.macro HANDLER HandleLabel
   sub sp,sp,#4     //给中断入口留地方
   stmfd sp!,{R0}  //要用到的R0 先放栈保护现场
   ldr R0 ,=HandleLabel //取地址
   ldr R0 ,[R0]       //中断服务程序入口给R0
   str R0 ,[sp,#4]    //ISR入栈
   ldmfd sp!, {R0,PC}//恢复R0并将PC指向ISR
.endm
调用时:
HandlerFIQ:  HANDKER HandleFIQ
HandlerIRQ:  HANDKER HandleIRQ
```

结尾:

初学ARM汇编将其分段整理成笔记供自己参考也供与大家学习，如有错误请大佬们直言指出，如果感觉有用那就点个赞留个言，谢谢观众老爷们的赏脸。

若想获得上述内容的PDF版本移步到GitHub下载。

地址: https://github.com/QianquanChina/Study-Notes

-----缱绻