



计算机组成原理实验报告

题目：从高级语言到机器语言

姓 名： 段欣然
专 业： 计算机科学与技术
年 级： 2020 级
学 号： 202011081033
任课教师： 王志春
完成日期： 2021 年 3 月 8 日

人工智能学院

一、 实验要求

对下面的 c 语言代码进行编译、汇编、链接（省略预编译阶段），最终生成可执行文件。对生成的可执行文件进行反汇编，观察汇编文件和可执行文件。

```
#include <stdio.h>
int main(){
    int a = 1, b = 2;
    int c = a + b;
    printf("%d\n", c);
    return 0;
}
```

二、 实验结果与分析

[根据实验的具体要求，列出子标题——描述实验结果和分析]

1. 实验步骤

- 1) gcc 预编译 lab1.c 并输出文件 lab1.i
- 2) 对 lab1.i 进行编译
- 3) 对 lab1.s 进行汇编
- 4) 生成可执行文件 lab1
- 5) 对可执行文件 lab1 反汇编生成目标文件 lab1.txt
- 6) 为便于观察 lab1.c 程序内容，对 lab1.o 反汇编生成文件 lab1_withoutLib.txt,

详细实验过程见下图。








```

PS E:\大二要努力\计组\作业\实验-> gcc --version
gcc.exe (MinGW.org GCC-6.3.0-1) 6.3.0
Copyright (C) 2016 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

PS E:\大二要努力\计组\作业\实验-> gcc -E lab1.c -o lab1.i
PS E:\大二要努力\计组\作业\实验-> gcc -S lab1.i
PS E:\大二要努力\计组\作业\实验-> gcc -c lab1.s
PS E:\大二要努力\计组\作业\实验-> gcc -d lab1 > lab1.txt
gcc.exe: fatal error: no input files
compilation terminated.
PS E:\大二要努力\计组\作业\实验-> gcc -o lab1 lab.o
gcc.exe: error: lab.o: No such file or directory
gcc.exe: fatal error: no input files
compilation terminated.
PS E:\大二要努力\计组\作业\实验-> gcc -o lab1 lab1.o
PS E:\大二要努力\计组\作业\实验-> objdump -d lab1 > lab1.txt
D:\MinGW\bin\objdump.exe: 'lab1': No such file
PS E:\大二要努力\计组\作业\实验-> objdump -d lab1.exe > lab1.txt
PS E:\大二要努力\计组\作业\实验-> objdump -d lab1.o > lab1_withoutLib.txt
PS E:\大二要努力\计组\作业\实验-> |

```

图表 1 具体实验过程

 lab1.c	2022/3/4 18:01	C Source	1 KB
 lab1.exe	2022/3/8 17:14	应用程序	40 KB
 lab1.i	2022/3/8 17:12	Preprocessed C/...	21 KB
 lab1.o	2022/3/8 17:13	O 文件	1 KB
 lab1.s	2022/3/8 17:12	Assembler Source	1 KB
 lab1.txt	2022/3/8 17:14	文本文档	365 KB
 lab1_withoutLib.txt	2022/3/8 17:15	文本文档	2 KB

图表 2 编译、汇编、反汇编后生成文件截图

```
lab1.txt - 记事本

文件  编辑  查看

lab1.exe:  file format pei-i386

Disassembly of section .text:

00401000 <.text>:
401000:  83 ec 1c          sub    $0x1c,%esp
401003:  8b 44 24 20       mov    0x20(%esp),%eax
401007:  8b 00             mov    (%eax),%eax
401009:  8b 00             mov    (%eax),%eax
40100b:  3d 91 00 00 c0    cmp    $0xc0000091,%eax
401010:  77 4e             ja     401060 <.text+0x60>
401012:  3d 8d 00 00 c0    cmp    $0xc000008d,%eax
401017:  73 60             jae    401079 <.text+0x79>
401019:  3d 05 00 00 c0    cmp    $0xc0000005,%eax
40101e:  0f 85 cc 00 00 00 jne    4010f0 <.text+0xf0>
401024:  c7 44 24 04 00 00 00 movl   $0x0,0x4(%esp)
40102b:  00
40102c:  c7 04 24 0b 00 00 00 movl   $0xb,(%esp)
401033:  e8 30 2a 00 00    call  403a68 <_signal>
401038:  83 f8 01          cmp    $0x1,%eax
40103b:  0f 84 48 01 00 00 je     401189 <.text+0x189>
```

图表 3lab1.txt 部分内容截图

```
lab1_withoutLib.txt - 记事本

文件  编辑  查看

lab1.o:  file format pe-i386

Disassembly of section .text:

00000000 <_main>:
  0:  55                push  %ebp
  1:  89 e5             mov   %esp,%ebp
  3:  83 e4 f0          and   $0xffffffff0,%esp
  6:  83 ec 20          sub   $0x20,%esp
  9:  e8 00 00 00 00    call  e <_main+0xe>
  e:  c7 44 24 1c 01 00 00  movl  $0x1,0x1c(%esp)
15:  00
16:  c7 44 24 18 02 00 00  movl  $0x2,0x18(%esp)
1d:  00
1e:  8b 54 24 1c       mov   0x1c(%esp),%edx
22:  8b 44 24 18       mov   0x18(%esp),%eax
26:  01 d0            add   %edx,%eax
28:  89 44 24 14       mov   %eax,0x14(%esp)
2c:  8b 44 24 14       mov   0x14(%esp),%eax
30:  89 44 24 04       mov   %eax,0x4(%esp)
34:  c7 04 24 00 00 00 00  movl  $0x0,(%esp)
3b:  e8 00 00 00 00    call  40 <_main+0x40>
40:  b8 00 00 00 00    mov   $0x0,%eax
45:  c9               leave
46:  c3               ret
47:  90               nop
```

图表 4lab1_withoutLib.txt 全部内容截图

2. 观察可执行目标文件的内容

```
lab1.exe.hexdump X
1823 00007100: 00 00 03 01 03 00 00 00 00 00 00 00 00 00 00 00 .....C.....
1824 000071e0: 00 00 00 00 00 00 00 00 00 00 E6 00 00 00 48 01 .....f...H.
1825 000071f0: 00 00 03 00 00 00 03 01 11 00 00 00 00 00 00 00 .....
1826 00007200: 00 00 00 00 00 00 00 00 00 00 2E 6A 63 72 00 00 .....jcr..
1827 00007210: 00 00 18 00 00 00 02 00 00 00 03 00 2E 66 69 6C .....fil
1828 00007220: 65 00 00 00 60 00 00 00 FE FF 00 00 67 01 6C 61 e...~...g.la
1829 00007230: 62 31 2E 63 00 00 00 00 00 00 00 00 00 00 00 00 b1.c.....
1830 00007240: 5F 66 61 69 6E 00 00 00 60 04 00 00 01 00 20 00 _main...`.....
1831 00007250: 02 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1832 00007260: 00 00 00 00 2E 74 65 78 74 00 00 00 60 04 00 00 .....text...`...
1833 00007270: 01 00 00 00 03 01 47 00 00 00 03 00 00 00 00 00 .....G.....
1834 00007280: 00 00 00 00 00 00 00 00 2E 64 61 74 61 00 00 00 .....data...
1835 00007290: 04 00 00 00 02 00 00 00 03 01 00 00 00 00 00 00 .....
1836 000072a0: 00 00 00 00 00 00 00 00 00 00 00 00 2E 62 73 73 .....bss
1837 000072b0: 00 00 00 00 20 00 00 00 05 00 00 00 03 01 00 00 .....
1838 000072c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1839 000072d0: 2E 72 64 61 74 61 00 00 64 00 00 00 03 00 00 00 .rdata..d.....
1840 000072e0: 03 01 04 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1841 000072f0: 00 00 00 00 00 00 00 00 E6 00 00 00 5C 01 00 00 .....f...\.
1842 00007300: 03 00 00 00 03 01 23 00 00 00 00 00 00 00 00 00 .....#.....
1843 00007310: 00 00 00 00 00 00 00 00 00 00 00 00 DC 00 00 00 .....\.
1844 00007320: 1C 01 00 00 04 00 00 00 03 01 38 00 00 00 01 00 .....8.....
1845 00007330: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1846 00007340: F1 00 00 00 B0 04 00 00 01 00 20 00 02 01 00 00 q...0.....
1847 00007350: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1848 00007360: 2E 74 65 78 74 00 00 00 B0 04 00 00 01 00 00 00 .text...0.....
1849 00007370: 03 01 9F 03 00 00 0F 00 00 00 00 00 00 00 00 00 .....
1850 00007380: 00 00 00 00 00 00 00 00 FB 00 00 00 50 08 00 00 .....{...P...
1851 00007390: 01 00 20 00 02 01 00 00 00 00 00 00 00 00 00 00 .....
1852 000073a0: 00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00 .....text...
1853 000073b0: 50 08 00 00 01 00 00 00 03 01 07 01 00 00 02 00 P.....
1854 000073c0: 00 00 00 00 00 00 00 00 00 00 00 00 2E 62 73 73 .....bss
1855 000073d0: 00 00 00 00 24 00 00 00 05 00 00 00 03 01 04 00 ....$.
1856 000073e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1857 000073f0: 00 00 00 00 10 01 00 00 60 09 00 00 01 00 20 00 .....`.....
1858 00007400: 02 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1859 00007410: 00 00 00 00 00 00 00 00 23 01 00 00 90 09 00 00 .....#.....
1860 00007420: 01 00 20 00 02 00 5F 5F 5F 6D 61 69 6E 00 E0 09 .....__main.`.
1861 00007430: 00 00 01 00 20 00 02 00 2E 74 65 78 74 00 00 00 .....text...
```

图表 5 可执行目标文件内容（十六进制表示）

3. 实验结果分析

1) 函数 printf()对应的机器代码段

```

00401460 <_main>:
401460: 55          push  %ebp
401461: 89 e5       mov   %esp,%ebp
401463: 83 e4 f0    and   $0xffffffff0,%esp
401466: 83 ec 20    sub   $0x20,%esp
401469: e8 72 05 00 00 call  4019e0 <__main>
40146e: c7 44 24 1c 01 00 00 movl  $0x1,0x1c(%esp)
401475: 00
401476: c7 44 24 18 02 00 00 movl  $0x2,0x18(%esp)
40147d: 00
40147e: 8b 54 24 1c mov   0x1c(%esp),%edx
401482: 8b 44 24 18 mov   0x18(%esp),%eax
401486: 01 d0       add   %edx,%eax
401488: 89 44 24 14 mov   %eax,0x14(%esp)
40148c: 8b 44 24 14 mov   0x14(%esp),%eax
401490: 89 44 24 04 mov   %eax,0x4(%esp)
401494: c7 04 24 64 50 40 00 movl  $0x405064,(%esp)
40149b: e8 e0 25 00 00 call  403a80 <_printf>
4014a0: b8 00 00 00 00 mov   $0x0,%eax
4014a5: c9         leave

```

图表 6 先定位到主函数调用 printf 函数

```

00403a80 <_printf>:
403a80: ff 25 d8 81 40 00      jmp   *0x4081d8
403a86: 90                    nop
403a87: 90                    nop

```

图表 7 printf 函数对应机器代码段

2) 源程序文件的内容和可执行目标文件的内容完全不同

预处理阶段，预处理器根据“#”字符开头的命令修改源程序，得到拓展名为.i 的文件。

编译阶段，编译器将文本文件翻译成 lab1.s，该文件为汇编语言程序，每条语句都描述了一条低级机器语言指令。

```

ASM lab1.s X
ASM lab1.s
1      .file   "lab1.c"
2      .def    __main; .scl 2; .type 32; .endef
3      .section .rdata,"dr"
4      LC0:
5      .ascii  "%d\\12\\0"
6      .text
7      .globl  _main
8      .def    _main; .scl 2; .type 32; .endef
9      _main:
10     LFB10:
11     .cfi_startproc
12     pushl   %ebp
13     .cfi_def_cfa_offset 8
14     .cfi_offset 5, -8
15     movl    %esp, %ebp
16     .cfi_def_cfa_register 5
17     andl    $-16, %esp
18     subl    $32, %esp
19     call    __main
20     movl    $1, 28(%esp)
21     movl    $2, 24(%esp)
22     movl    28(%esp), %edx
23     movl    24(%esp), %eax
24     addl    %edx, %eax
25     movl    %eax, 20(%esp)
26     movl    20(%esp), %eax
27     movl    %eax, 4(%esp)
28     movl    $LC0, (%esp)
29     call    _printf
30     movl    $0, %eax
31     leave
32     .cfi_restore 5
33     .cfi_def_cfa 4, 4
34     ret
35     .cfi_endproc
36     LFE10:
37     .ident   "GCC: (MinGW.org GCC-6.3.0-1) 6.3.0"

```

图表 8lab1.s 部分示例

汇编阶段，汇编器将汇编语言翻译成机器语言指令，并将结果保存在二进制文件 lab1.o 中。

链接阶段，把标准 C 库中的函数（例如 printf）并入文件中，生成可执行文件。

3) 不同的编译器或操作系统生成可执行目标文件

为便于比较，在此仅展示用 lab1.o 反汇编的主函数部分。

a) Win10+ GCC: (MinGW.org GCC-6.3.0-1) 6.3.0

```
lab1.o:  file format pe-i386

Disassembly of section .text:

00000000 <_main>:
0:  55                push  %ebp
1:  89 e5             mov   %esp,%ebp
3:  83 e4 f0          and   $0xffffffff0,%esp
6:  83 ec 20          sub   $0x20,%esp
9:  e8 00 00 00 00    call  e <_main+0xe>
e:  c7 44 24 1c 01 00 00 movl  $0x1,0x1c(%esp)
15: 00
16: c7 44 24 18 02 00 00 movl  $0x2,0x18(%esp)
1d: 00
1e: 8b 54 24 1c       mov   0x1c(%esp),%edx
22: 8b 44 24 18       mov   0x18(%esp),%eax
26: 01 d0             add   %edx,%eax
28: 89 44 24 14       mov   %eax,0x14(%esp)
2c: 8b 44 24 14       mov   0x14(%esp),%eax
30: 89 44 24 04       mov   %eax,0x4(%esp)
34: c7 04 24 00 00 00 00 movl  $0x0,(%esp)
3b: e8 00 00 00 00    call  40 <_main+0x40>
40: b8 00 00 00 00    mov   $0x0,%eax
45: c9               leave
46: c3               ret
47: 90               nop
```

图表 9Win10 系统使用 GCC: (MinGW.org GCC-6.3.0-1) 6.3.0 编译

b) Win10+gcc (tdm64-1) 10.3.0

```

lab1.txt - 记事本
文件 编辑 查看

Disassembly of section .text:

0000000000000000 <printf>:
  0: 55          push %rbp
  1: 53          push %rbx
  2: 48 83 ec 38  sub $0x38,%rsp
  6: 48 8d 6c 24 30 lea 0x30(%rsp),%rbp
  b: 48 89 4d 20    mov %rcx,0x20(%rbp)
  f: 48 89 55 28    mov %rdx,0x28(%rbp)
 13: 4c 89 45 30    mov %r8,0x30(%rbp)
 17: 4c 89 4d 38    mov %r9,0x38(%rbp)
 1b: 48 8d 45 28    lea 0x28(%rbp),%rax
 1f: 48 89 45 f0    mov %rax,-0x10(%rbp)
 23: 48 8b 5d f0    mov -0x10(%rbp),%rbx
 27: b9 01 00 00 00 mov $0x1,%ecx
 2c: 48 8b 05 00 00 00 00 mov 0x0(%rip),%rax # 33 <printf+0x33>
 33: ff d0        call *%rax
 35: 49 89 d8      mov %rbx,%r8
 38: 48 8b 55 20    mov 0x20(%rbp),%rdx
 3c: 48 89 c1      mov %rax,%rcx
 3f: e8 00 00 00 00 call 44 <printf+0x44>
 44: 89 45 fc      mov %eax,-0x4(%rbp)
 47: 8b 45 fc      mov -0x4(%rbp),%eax
 4a: 48 83 c4 38    add $0x38,%rsp
 4e: 5b          pop %rbx
 4f: 5d          pop %rbp
 50: c3          ret

0000000000000051 <main>:
 51: 55          push %rbp
 52: 48 89 e5      mov %rsp,%rbp
 55: 48 83 ec 30    sub $0x30,%rsp
 59: e8 00 00 00 00 call 5e <main+0xd>
 5e: c7 45 fc 01 00 00 00 movl $0x1,-0x4(%rbp)

```

图表 10Win10 系统使用 gcc (tdm64-1) 10.3.0 编译

c) Mac+Apple clang version 13.0.0 (clang-1300.0.29.30)

```
lab1.txt
|
lab1.o: file format mach-o 64-bit x86-64

Disassembly of section __TEXT,__text:

0000000000000000 <_main>:
  0: 55                                pushq   %rbp
  1: 48 89 e5                          movq    %rsp, %rbp
  4: 48 83 ec 10                       subq    $16, %rsp
  8: c7 45 fc 00 00 00 00             movl    $0, -4(%rbp)
  f: c7 45 f8 01 00 00 00             movl    $1, -8(%rbp)
 16: c7 45 f4 02 00 00 00             movl    $2, -12(%rbp)
1d: 8b 45 f8                          movl    -8(%rbp), %eax
20: 03 45 f4                          addl    -12(%rbp), %eax
23: 89 45 f0                          movl    %eax, -16(%rbp)
26: 8b 75 f0                          movl    -16(%rbp), %esi
29: 48 8d 3d 0f 00 00 00             leaq    15(%rip), %rdi # 3f <_main+0x3f>
30: b0 00                            movb    $0, %al
32: e8 00 00 00 00                   callq   0x37 <_main+0x37>
37: 31 c0                            xorl    %eax, %eax
39: 48 83 c4 10                       addq    $16, %rsp
3d: 5d                                popq    %rbp
3e: c3                                retq

*****
```

图表 11MAC 系统下使用 Apple clang version 13.0.0 (clang-1300.0.29.30) 编译

三、 实验小结

高级语言编写程序后运行需要经过四个阶段：预处理阶段、编译阶段、预处理阶段，汇编阶段和链接阶段。以下用 lab1.c 为例分析四个阶段的作用。

预处理器根据“#”字符开头的命令修改源程序，得到拓展名为.i 的文件。例如 lab1.c 中的#include<stdio.h>指令告诉预处理器读取系统呕吐文件内容，并插入程序文本中，得到 lab1.i。

编译阶段，编译器将文本文件翻译成 lab1.s，该文件为汇编语言程序，每条语句都描述了一条低级机器语言指令。不同的编译器汇编语句有略微差异，但是汇编语言是通用的计算机语言，所以不会出现较大差别，描述的指令均为相同的操作。

汇编阶段，汇编器将汇编语言翻译成机器语言指令，并将结果保存在二进制文件 lab1.o 中。lab1.o 的字节编码是机器语言指令而不是字符，所以打开时呈现

乱码。

链接阶段，把标准 C 库中的函数（例如 printf）并入文件中，生成可执行目标文件。可执行文件加载到存储器后，由系统负责执行。