

定义你的编译器功能 & 汇编编程

潘宇

September 2019

目录

1	MASM32 Editor 的安装与使用	3
1.1	MASM32 Editor 的安装	3
1.2	MASM32 Editor 的使用	3
2	定义你的编译器功能	5
2.1	作业要求	5
2.2	作业解析	6
2.2.1	上下文无关文法	6
2.2.2	CFG 描述 C 语言特性举例	7
3	汇编编程	8
3.1	作业要求	8
3.2	作业解析	8

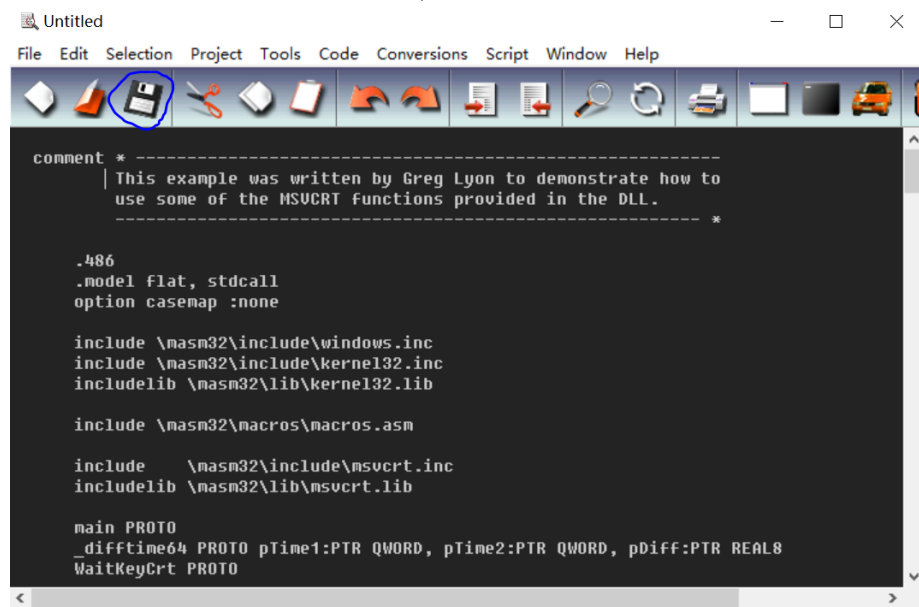
1 MASM32 Editor 的安装与使用

1.1 MASM32 Editor 的安装

MASM32 Editor 的安装程序位于开发工具文件夹下的安装程序文件夹下，其中 install.exe 为 MASM32 Editor 的安装程序，直接运行安装即可。

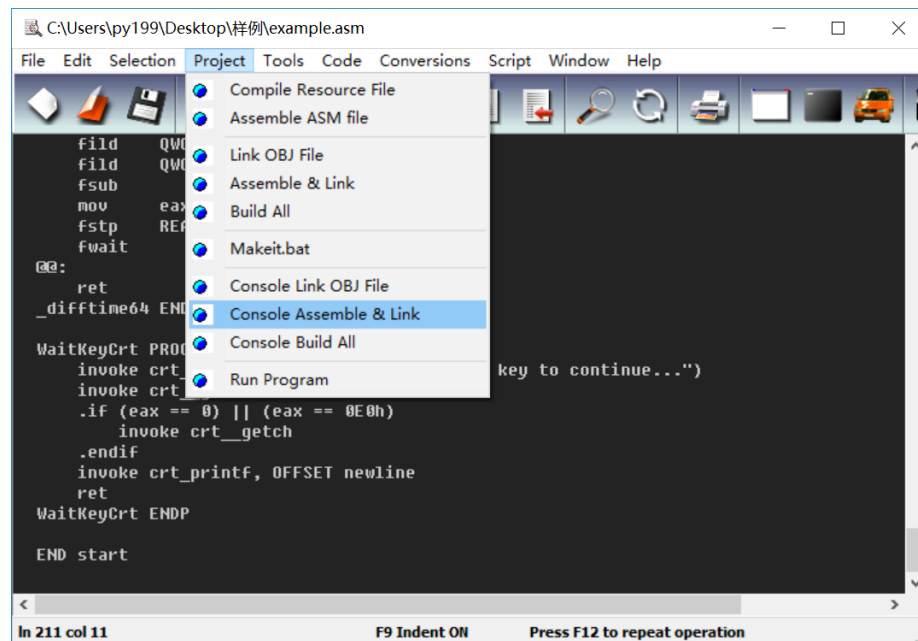
1.2 MASM32 Editor 的使用

(1) 打开 MASM32 Editor，编写汇编代码 (MASM32 文件夹中包含很多例程，这里我使用 \masm32\tools\makecimp\vcrtdemo 目录中 msvcrt 的例程，同学们可任意选取例程尝试)，点击如下图所示的保存按钮：



第一次保存需要输入文件名，注意要指定扩展名.asm，比如：example.asm

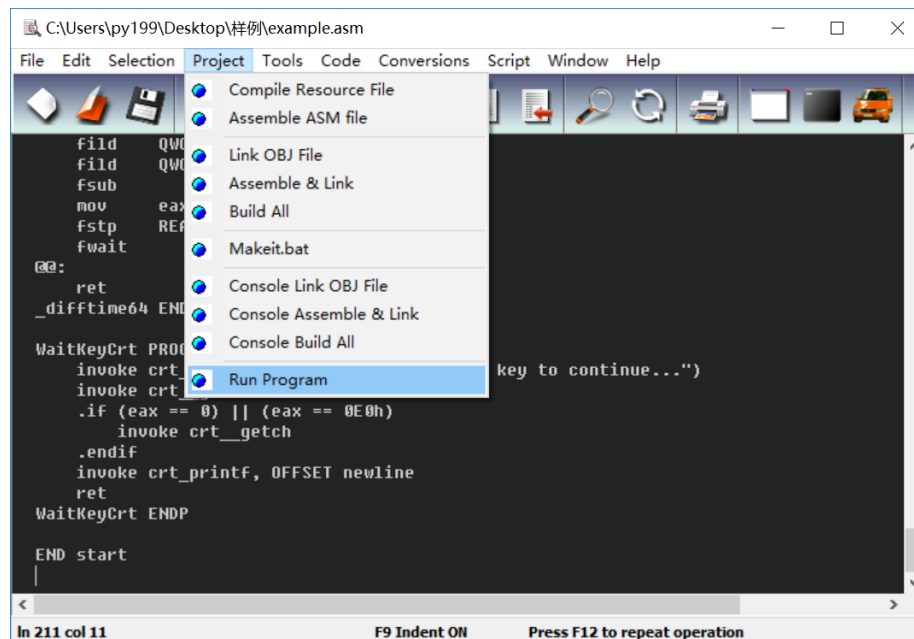
(2) 点击 Project->Console Assemble &Link 进行汇编链接：




如果成功，会生成对应的 obj 文件 (example.obj) 和 exe 文件 (example.exe)，如下图所示：

example.asm	2019/9/1 13:48	ASM source file	8 KB
example.exe	2019/9/1 13:48	应用程序	4 KB
example.obj	2019/9/1 13:48	3D Object	5 KB

(3) 点击 Project->Run Program 运行生成的可执行文件，也可以直接运行对应的 exe 文件 (example.exe)



样例运行结果如下:

 C:\Users\py199\Desktop\样例\example.exe

```
64-bit Date and Time Functions - C Run-time Library
OS time:                13:48:51
OS date:                09/01/19
Time in seconds since UTC 1/1/1970: 18031990699732992
Time and date string:    Sun Sep 01 13:48:51 2019
Coordinated universal time: Sun Sep 01 05:48:51 2019
12-hour time:           01:48:51 PM
Plus milliseconds:      588
Zone difference in hours from UTC: 71582780
Time zone name:         中国标准时间
Daylight savings:       False
Today is:               Sunday, September 01, 2019
Christmas this year:    Wednesday, December 25, 2019
Days until Christmas:   115

Press any key to continue..._
```

2 定义你的编译器功能

2.1 作业要求

你所使用的编译器支持哪些主要的 C (C++) 语言特性? 在此基础上定义你的编译器支持的 C 语言子集——学习教材第 2 章及第 2 章讲义中的

2.2 节，用上下文无关文法描述你的 C 语言子集。

2.2 作业解析

这一部分作业的主要要求是了解你的编译器所支持的 C (C++) 语言特性，如支持何种数据类型 (int,double 等), 支持变量声明, 赋值语句, 复合语句, if 分支语句, 以及 while/for 循环, 支持算术运算 (加减乘除按位与或等)、逻辑运算 (逻辑与或等)、关系运算 (不等等于大于小于等), 支持函数, 数组指针等等。从中选取重要的部分定义为你编译器功能, 使用上下文无关文法描述你所选取的 C 语言子集。

2.2.1 上下文无关文法

上下文无关文法是一种用于描述程序设计语言语法的表示方式。一般来说, 一个上下文无关文法 (context-free grammar) 由四个元素组成:

(1) 一个终结符号集合 V_T 它们有时也称为“词法单元”。终结符号是该文法所定义的语言的基本符号的集合。

(2) 一个非终结符号集合 V_N 它们有时也称为“语法变量”。每个非终结符号表示一个终结符号串的集合。

(3) 一个产生式集合 P , 其中每个产生式包括一个称为产生式头或左部的非终结符号, 一个箭头, 和一个称为产生式体或右部的由终结符号及非终结符号组成的序列。产生式主要用来表示某个构造的某种书写形式。如果产生式头非终结符号代表一个构造, 那么该产生式体就代表了该构造的一种书写方式。

(4) 指定一个非终结符号为开始符号 S 。

因此, 上下文无关文法可以通过 (V_T, V_N, P, S) 这个四元式定义。在描述文法时, 我们将数位、符号和黑体字符串看作终结符号, 将斜体字符串看作非终结符号, 以同一个非终结符号为头部的多个产生式的右部可以放在一起表示, 不同的右部之间用符号 | 分隔。

上下文无关文法无论是对课程内容的学习还是之后实践上机作业都是十分重要的, 希望同学们能够认真学习并扎实掌握。

2.2.2 CFG 描述 C 语言特性举例

1. 变量声明

$$type \rightarrow \text{int} \mid \text{float} \mid \text{double} \mid \text{char}$$

$$idlist \rightarrow idlist, id \mid id$$

$$decl \rightarrow type \ idlist$$

其中 id 表示标识符, type 代表变量类型, idlist 代表标识符列表, decl 代表声明语句。

2. 赋值语句

$$digit \rightarrow number \ digit$$

$$number \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

$$unary - expr \rightarrow digit \mid id$$

$$assign - expr \rightarrow unary - expr = assign - expr \mid logical - expr$$

其中 num 代表数字字符, digit 代表数字, logical-expr 为逻辑表达式, unary-expr 为一元表达式, assign-expr 为赋值表达式, 这里最后一个产生式第二个右部为逻辑表达式的原因是因为逻辑表达式的逻辑与和逻辑或优先级高于赋值表达式但却低于关系表达式的比较和算术表达式的各种运算。同学们写 CFG 时需格外注意优先级对 CFG 所带来的影响, 经典例子为加减和乘除的 CFG(可查看龙书 P30)

3. 循环语句及分支语句

$$stmt \rightarrow \text{if} (expr) \ stmt \ \text{else} \ stmt$$

$$stmt \rightarrow \text{while} (expr) \ stmt$$

$$stmt \rightarrow \text{for} (expr; expr; expr) \ stmt$$

其中 stmt 为语句, expr 为表达式。

4. 函数定义

$$funcdef \rightarrow type \ funcname(paralist) \ stmt$$

$$paralist \rightarrow paralist, parafdef \mid parafdef \mid \epsilon$$

$$parafdef \rightarrow type \ id$$

其中 paralist 代表参数列表, parafdef 代表参数声明, funcname 代表函数名, funcdef 代表函数声明语句。

3 汇编编程

3.1 作业要求

对某个 C 程序（如“预备工作 1”给出的阶乘或斐波那契），编写等价的汇编程序，用汇编器生成可执行程序，调试通过、能正常运行。（注意：汇编程序中不可直接使用 masm32 所支持的 while 语句）

3.2 作业解析

这一部分需要一些简单基础的汇编知识，如 mov,add 等指令的使用等，请同学们温习相关汇编知识。接下来我将通过一个程序的例子来说明如何使用 masm32 editor 编写汇编程序。

样例问题描述如下：

计算前 n 个自然数 (0,1,2...) 之和并输出结果。

样例的源代码如下：

```
1 main()
2 {
3     int i, n, f;
4     cin >> n;
5     i = 1;
6     f = 0;
7     while (i < n)
8     {
9         f = f + i;
10        i = i + 1;
11    }
12    cout << f << endl;
13 }
```

对应编写的 masm32 的汇编程序如下：

```
1 .486
2 .model flat, stdcall
3
4 option casemap:none
5
6 includelib \masm32\lib\kernel32.lib
7 includelib \masm32\lib\user32.lib
8 includelib \masm32\lib\gdi32.lib
9 includelib \masm32\lib\msvcrt.lib
10 includelib \masm32\lib\masm32.lib
11
12
13 include \masm32\include\kernel32.inc
14 include \masm32\include\user32.inc
15 include \masm32\include\gdi32.inc
16 include \masm32\include\windows.inc
17 include \masm32\include\msvcrt.inc
18 include \masm32\include\masm32.inc
19 include \masm32\macros\macros.asm
```



```

20 main PROTO
21 WaitKeyCrt PROTO
22
23 .data
24     n dd 0
25     f dd 0
26     i dd 1
27     newline BYTE 13, 10, 0
28 .code
29 start:
30     invoke main
31     invoke WaitKeyCrt
32     invoke crt__exit, 0
33
34 main PROC uses edi
35     mov eax, sval(input("Enter a number : "))
36     mov n,eax
37     jmp L3
38     L2:
39         mov eax,f
40         add eax,i
41         mov f,eax
42         inc i
43     L3:
44         mov ecx,i
45         cmp ecx,n
46         jl L2
47         invoke crt_printf, SADD("The result is: %d"), f
48         ret
49 main ENDP
50 WaitKeyCrt PROC
51     invoke crt_printf, SADD(13,10,"Press any key to continue...")
52     invoke crt__getch
53     .if (eax == 0) || (eax == 0E0h)
54         invoke crt__getch
55     .endif
56     invoke crt_printf, OFFSET newline
57     ret
58 WaitKeyCrt ENDP
59 end start

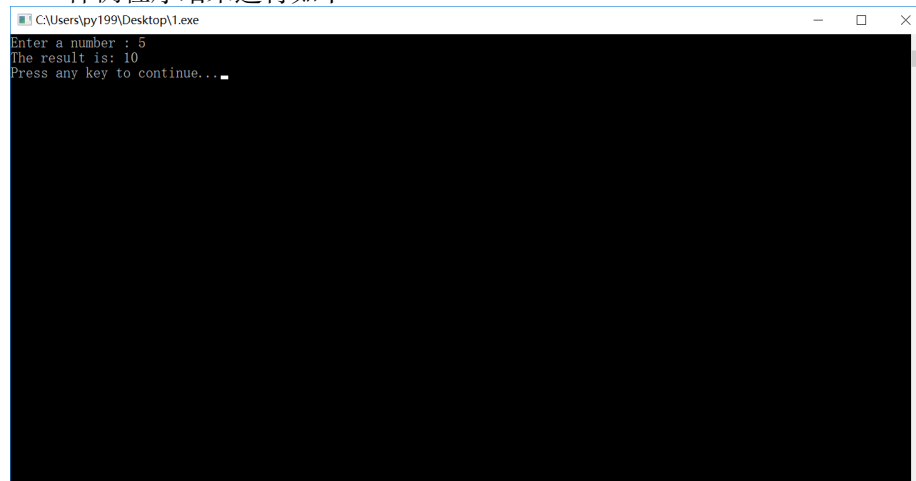
```

除 24-26 行和 35-47 行以外为我写的 masm32 汇编程序框架，第一行.486 代表使用 486 处理器，第二行.model flat,stdcall 代表使用平坦内存模式和 stdcall 调用习惯。第四行 option casemap:none 代表控制字符的映射为大写。第 6 到 19 行为所需包含的文件。第 20 行和 21 行代表代码段将有 main 和 WaitKeyCrt 两个子程序部分。23 行.data 是变量定义部分，之后定义各种变量及初值。其中 dd 代表双字类型变量，BYTE 代表字节类型变量。其中 newline 初值的 13,10,0 代表“\r\n\0”即换新的一行。第 28 行.code 代表之后是代码段。第 29 行 start 和第 60 行 end start 之间代表运行的代码。第 30-32 行代表调用的三个函数：main 函数，按任意键退出的等待按键 WaitKeyCrt 函数，退出程序函数。第 51-59 行为 WaitKeyCrt 函数的定义部分，51 行调用 crt_printf 函数屏幕输出，53-56 行调用 crt_getch 函数，对于方向键和功能键需要调用两次，**注意一定是两次！第一次返回 0 或**

0xE0, 第二次返回实际的键值, 其余普通键调用一次即可返回实际键值。第 57 行调用 `crt_printf` 函数换行。第 58 行 `ret` 指令是每个子程序部分所必须的, 放在子程序的结尾, 当子程序执行完后, 靠该指令返回主程序。框架部分介绍完成, 其中 `WaitKeyCrt` 函数并不是必须的, 同学们可以不写。

对于这个样例, 需要 `n`, `f`, `i` 三个变量, 初值分别为 0, 0, 1 因此在 24-26 行数据声明部分声明。之后源程序首先需要输入 `n` 的值, 对应 35-36 行, 调用 `sval` 函数获取输入。获取输入的 `n` 值之后跳到 L3 部分进行 `while` 的条件判断, 使用 `cmp` 指令比较 `i` 和 `n` 的值, `jl` 代表小于则跳转, 跳转到 L2 部分, 对 `f` 进行累加, `inc` 指令代表自加一。跳出 `while` 循环后调用 `crt_printf` 函数输出结果。

样例程序结果运行如下:



```
C:\Users\py199\Desktop\1.exe
Enter a number : 5
The result is: 10
Press any key to continue...
```

阶乘及斐波那契程序可仿照样例完成, 鼓励同学们完成更多复杂的程序, 以增进自己对汇编知识的掌握。