北京师范大学2018-2019学年第一学期 “汇编程序设计”

上机作业九

姓 名	学 号	日 期
刘源	201611210134	2018.11.28

实验目的

- **理解调条件寄存器的意义。**
- **掌握使用带进位的加法实现高位宽数据的计算的方法。**

实验总结：

这次试验了解了addl与adcl的区别，并且通过查看C语言的汇编了解了32位的汇编在处理128位大数加法的处理方法。

实验题目：实现128位整数加法

步骤1：分析main.cpp中的结构体传参的过程

```
    .file   "main.cpp"
.lcomm __ZStL8__ioinit,1,1
    .section    .text$_ZSt7setbasei,"x"
    .linkonce discard
    .globl  __ZSt7setbasei
    .def    __ZSt7setbasei; .scl    2;  .type   32; .endef
__ZSt7setbasei:
    pushl   %ebp
    movl    %esp, %ebp
    movl    8(%ebp), %eax
    popl    %ebp
    ret
    .section    .text$_ZSt4setwi,"x"
    .linkonce discard
    .globl  __ZSt4setwi
    .def    __ZSt4setwi;    .scl    2;  .type   32; .endef
__ZSt4setwi:
    pushl   %ebp
    movl    %esp, %ebp
    movl    8(%ebp), %eax
    popl    %ebp
    ret
    .text
    .globl  __Z11printint1286int128
    .def    __Z11printint1286int128;    .scl    2;  .type   32; .endef
__Z11printint1286int128:
    pushl   %ebp
    movl    %esp, %ebp
    pushl   %edi
    pushl   %esi
    pushl   %ebx

    subl    $28, %esp
    movl    20(%ebp), %ebx
    movl    $48, (%esp)
    call    __ZSt7setfillIcESt8_SetfillIT_ES1_
    movl    %eax, %esi
    movl    $8, (%esp)
    call    __ZSt4setwi
    movl    %eax, %edi
    movl    $16, (%esp)
    call    __ZSt7setbasei
    movl    %eax, 4(%esp)
    movl    $__ZSt4cout, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_Setbase
    movl    %edi, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St5_Setw
    movl    %esi, %edx
    movb    %dl, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_SetfillIS3_E
    movl    %ebx, (%esp)
    movl    %eax, %ecx
    call    __ZNSolsEi

    subl    $4, %esp
    movl    16(%ebp), %ebx
    movl    $48, (%esp)
    call    __ZSt7setfillIcESt8_SetfillIT_ES1_
    movl    %eax, %esi
    movl    $8, (%esp)
    call    __ZSt4setwi
    movl    %eax, %edi
    movl    $16, (%esp)
    call    __ZSt7setbasei
    movl    %eax, 4(%esp)
    movl    $__ZSt4cout, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_Setbase
    movl    %edi, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St5_Setw
    movl    %esi, %edx
    movb    %dl, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_SetfillIS3_E
    movl    %ebx, (%esp)
    movl    %eax, %ecx
    call    __ZNSolsEi

    subl    $4, %esp
    movl    12(%ebp), %ebx
    movl    $48, (%esp)
    call    __ZSt7setfillIcESt8_SetfillIT_ES1_
    movl    %eax, %esi
    movl    $8, (%esp)
    call    __ZSt4setwi
    movl    %eax, %edi
    movl    $16, (%esp)
    call    __ZSt7setbasei
    movl    %eax, 4(%esp)
    movl    $__ZSt4cout, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_Setbase
    movl    %edi, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St5_Setw
    movl    %esi, %edx
    movb    %dl, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_SetfillIS3_E
    movl    %ebx, (%esp)
    movl    %eax, %ecx
    call    __ZNSolsEi

    subl    $4, %esp
    movl    8(%ebp), %ebx
    movl    $48, (%esp)
    call    __ZSt7setfillIcESt8_SetfillIT_ES1_
    movl    %eax, %esi
    movl    $8, (%esp)
    call    __ZSt4setwi
    movl    %eax, %edi
    movl    $16, (%esp)
    call    __ZSt7setbasei
    movl    %eax, 4(%esp)
    movl    $__ZSt4cout, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_Setbase
    movl    %edi, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St5_Setw
    movl    %esi, %ecx
    movb    %cl, 4(%esp)
    movl    %eax, (%esp)
    call    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_SetfillIS3_E
    movl    %ebx, (%esp)
    movl    %eax, %ecx
    call    __ZNSolsEi
    subl    $4, %esp
    nop
    leal    -12(%ebp), %esp
    popl    %ebx
    popl    %esi
    popl    %edi
    popl    %ebp
    ret
    .def    ___main;    .scl    2;  .type   32; .endef
    .globl  _main
    .def    _main;  .scl    2;  .type   32; .endef
_main:
    pushl   %ebp
    movl    %esp, %ebp
    andl    $-16, %esp
    subl    $96, %esp           #
    call    ___main
    movl    $1, 80(%esp)        # 实参a初始化
    movl    $0, 84(%esp)
    movl    $0, 88(%esp)
    movl    $0, 92(%esp)
    movl    $2, 64(%esp)        # 实参b初始化
    movl    $0, 68(%esp)
    movl    $0, 72(%esp)
    movl    $0, 76(%esp)
    leal    48(%esp), %eax      # 将实参c的地址，放在32(%esp)这个地址
    movl    %eax, 32(%esp)
    movl    64(%esp), %eax      # 形参b初始化
    movl    %eax, 16(%esp)
    movl    68(%esp), %eax
    movl    %eax, 20(%esp)
    movl    72(%esp), %eax
    movl    %eax, 24(%esp)
    movl    76(%esp), %eax
    movl    %eax, 28(%esp)
    movl    80(%esp), %eax      # 形参a初始化
    movl    %eax, (%esp)
    movl    84(%esp), %eax
    movl    %eax, 4(%esp)
    movl    88(%esp), %eax
    movl    %eax, 8(%esp)
    movl    92(%esp), %eax
    movl    %eax, 12(%esp)
    call    __Z6add1286int128S_RS_
    movl    48(%esp), %eax
    movl    %eax, (%esp)
    movl    52(%esp), %eax
    movl    %eax, 4(%esp)
    movl    56(%esp), %eax
    movl    %eax, 8(%esp)
    movl    60(%esp), %eax
    movl    %eax, 12(%esp)
    call    __Z11printint1286int128
    movl    $0, %eax
    leave
    ret
    .section    .text$_ZSt7setfillIcESt8_SetfillIT_ES1_,"x"
    .linkonce discard
    .globl  __ZSt7setfillIcESt8_SetfillIT_ES1_
    .def    __ZSt7setfillIcESt8_SetfillIT_ES1_; .scl    2;  .type   32; .endef
__ZSt7setfillIcESt8_SetfillIT_ES1_:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $4, %esp
    movl    8(%ebp), %eax
    movb    %al, -4(%ebp)
    movzbl  -4(%ebp), %eax
    leave
    ret
    .text
    .def    ___tcf_0;   .scl    3;  .type   32; .endef
___tcf_0:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $8, %esp
    movl    $__ZStL8__ioinit, %ecx
    call    __ZNSt8ios_base4InitD1Ev
    leave
    ret
    .def    __Z41__static_initialization_and_destruction_0ii;   .scl    3;  .type   32;
.endef
__Z41__static_initialization_and_destruction_0ii:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $24, %esp
    cmpl    $1, 8(%ebp)
    jne L12
    cmpl    $65535, 12(%ebp)
    jne L12
    movl    $__ZStL8__ioinit, %ecx
    call    __ZNSt8ios_base4InitC1Ev
    movl    $___tcf_0, (%esp)
    call    _atexit
L12:
    leave
    ret
    .def    __GLOBAL__sub_I__Z11printint1286int128; .scl    3;  .type   32; .endef
__GLOBAL__sub_I__Z11printint1286int128:
    pushl   %ebp
    movl    %esp, %ebp
    subl    $24, %esp
    movl    $65535, 4(%esp)
    movl    $1, (%esp)
    call    __Z41__static_initialization_and_destruction_0ii
    leave
    ret
    .section    .ctors,"w"
    .align 4
    .long   __GLOBAL__sub_I__Z11printint1286int128
    .ident  "GCC: (tdm-1) 4.9.2"
    .def    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_Setbase;
.scl    2;  .type   32; .endef
    .def    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St5_Setw;   .scl
2;  .type   32; .endef
    .def    __ZStlsIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_St8_SetfillIS3_E;
.scl    2;  .type   32; .endef
    .def    __ZNSolsEi; .scl    2;  .type   32; .endef
    .def    __Z6add1286int128S_RS_; .scl    2;  .type   32; .endef
    .def    __ZNSt8ios_base4InitD1Ev;   .scl    2;  .type   32; .endef
    .def    __ZNSt8ios_base4InitC1Ev;   .scl    2;  .type   32; .endef
    .def    _atexit;    .scl    2;  .type   32; .endef
```

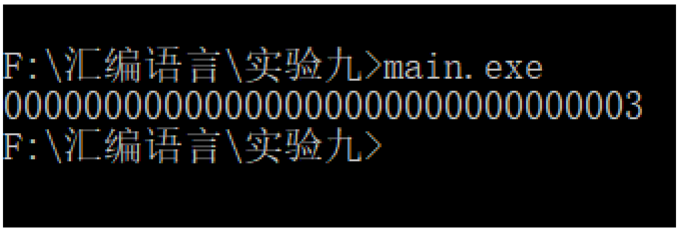
步骤2：自己手动编写一个add128.s文件，实现两个128位的整数相加的功能。

```
    .file   "add128.cpp"
    .text
    .globl  __Z6add1286int128S_RS_
    .def    __Z6add1286int128S_RS_; .scl    2;  .type   32; .endef
__Z6add1286int128S_RS_:
    pushl   %ebp
    movl    %esp, %ebp
    movl    8(%ebp), %edx
    movl    24(%ebp), %eax
    addl    %eax, %edx
    movl    40(%ebp), %eax
    movl    %edx, (%eax)
    movl    12(%ebp), %edx
    movl    28(%ebp), %eax
    adcl    %eax, %edx
    movl    40(%ebp), %eax
    movl    %edx, 4(%eax)
    movl    16(%ebp), %edx
    movl    32(%ebp), %eax
    adcl    %eax, %edx
    movl    40(%ebp), %eax
    movl    %edx, 8(%eax)
    movl    20(%ebp), %edx
    movl    36(%ebp), %eax
    adcl    %eax, %edx
    movl    40(%ebp), %eax
```

```
        movl    %edx, 12(%eax)
        nop
        popl    %ebp
        ret
```

步骤3：然后将自己编写的add128.s 生成目标文件，并和原来的main.o进行连接生成可执行文件。

执行结果



```
F:\汇编语言\实验九>main.exe
00000000000000000000000000000003
F:\汇编语言\实验九>
```