COSE222, COMP212 Computer Architecture
Assignment #1

Solutions

Prepared by Sunhee Kong

2. Using the C program in Table 1, go through the compilation steps we studied in the class. After each step, print out the outcome.

- Native compilation to generate x86 machine code under Cygwin

  - Preprocessing

```
$ cpp compare.c > compare.i
```

compare.i

```
# 1 "compare.c"
# 1 "<built-in>"
# 1 "<command-line>"
# 1 "compare.c"


int compare()
{
 int a, b, c;
 a = ((b) < (c) ? (b) : (c));;
 return a;
}

```

  - Compilation

```
$ gcc -S compare.i
```

compare.s

```
	.file	"compare.c"
	.text
	.globl	_compare
	.def	_compare;	.scl	2;	.type	32;	.endef
_compare:
LFB0:
	.cfi_startproc
	pushl	%ebp
	.cfi_def_cfa_offset 8
	.cfi_offset 5, -8
	movl	%esp, %ebp
	.cfi_def_cfa_register 5
	subl	$16, %esp
	movl	-8(%ebp), %eax
	cmpl	%eax, -4(%ebp)
	cmovle	-4(%ebp), %eax
	movl	%eax, -12(%ebp)
	movl	-12(%ebp), %eax
	leave
	.cfi_restore 5
	.cfi_def_cfa 4, 4
	ret
	.cfi_endproc
LFE0:
	.ident	"GCC: (GNU) 4.8.3"

```

- Assembler (Disassembled result)

```
$ as compare.s -o compare.o
```

```
$ objdump -SD compare.o > compare.dump
```

compare.dump

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


Disassembly of section .text:

00000000 <_compare>:
   0:   55                      push   %ebp
   1:   89 e5                   mov    %esp,%ebp
   3:   83 ec 10                sub    $0x10,%esp
   6:   8b 45 f8                mov    -0x8(%ebp),%eax
   9:   39 45 fc                cmp    %eax,-0x4(%ebp)
   c:   0f 4e 45 fc             cmovle -0x4(%ebp),%eax
  10:   89 45 f4                mov    %eax,-0xc(%ebp)
  13:   8b 45 f4                mov    -0xc(%ebp),%eax
  16:   c9                      leave
  17:   c3                      ret

Disassembly of section .rdata$zzz:

00000000 <.rdata$zzz>:
   0:   47                      inc    %edi
   1:   43                      inc    %ebx
   2:   43                      inc    %ebx
   3:   3a 20                   cmp    (%eax),%ah
   5:   28 47 4e                sub    %al,0x4e(%edi)
   8:   55                      push   %ebp
   9:   29 20                   sub    %esp,(%eax)
   b:   34 2e                   xor    $0x2e,%al
   d:   38 2e                   cmp    %ch,(%esi)
   f:   33 00                   xor    (%eax),%eax
  11:   00 00                   add    %al,(%eax)
    ...

Disassembly of section .eh_frame:

00000000 <.eh_frame>:
   0:   14 00                   adc    $0x0,%al
   2:   00 00                   add    %al,(%eax)
   4:   00 00                   add    %al,(%eax)
   6:   00 00                   add    %al,(%eax)
   8:   01 7a 52                add    %edi,0x52(%edx)
   b:   00 01                   add    %al,(%ecx)
   d:   7c 08                   jl     17 <.eh_frame+0x17>
   f:   01 1b                   add    %ebx,(%ebx)
  11:   0c 04                   or     $0x4,%al
  13:   04 88                   add    $0x88,%al
  15:   01 00                   add    %eax,(%eax)
  17:   00 1c 00                add    %bl,(%eax,%eax,1)
  1a:   00 00                   add    %al,(%eax)
  1c:   1c 00                   sbb    $0x0,%al
  1e:   00 00                   add    %al,(%eax)
  20:   04 00                   add    $0x0,%al
  22:   00 00                   add    %al,(%eax)
  24:   18 00                   sbb    %al,(%eax)
  26:   00 00                   add    %al,(%eax)
  28:   00 41 0e                add    %al,0xe(%ecx)
  2b:   08 85 02 42 0d 05       or     %al,0x50d4202(%ebp)
  31:   54                      push   %esp
  32:   c5 0c 04                lds    (%esp,%eax,1),%ecx
  35:   04 00                   add    $0x0,%al
    ...
```

- Cross-compilation to generate MIPS machine code using Eclipse environment

Makefile

```
HOME=c:/cygwin/home/esca
MIPSBIN=$(HOME)/mips-elf/bin
AS=$(MIPSBIN)/mips-elf-as
LD=$(MIPSBIN)/mips-elf-ld
CC=$(MIPSBIN)/mips-elf-gcc
CPP=$(MIPSBIN)/mips-elf-cpp
OBJDUMP=$(MIPSBIN)/mips-elf-objdump
OBJCOPY=$(MIPSBIN)/mips-elf-objcopy
#   ASFLAGS= -Wall -O2 -g
    ASFLAGS= -g
    LDFLAGS= -N -X -Ttestvec.lds
    CCFLAGS= -c -g

    all: testvec

    testvec: testvec.o add.o
    $(LD) $(LDFLAGS)  testvec.o add.o -o testvec          ← Linking
    $(OBJDUMP) -xS  testvec > testvec.dump                ← Disassembling the binary
    $(OBJCOPY) -O  binary testvec testvec.bin
    ./bin2hex.perl > testvec.hex
    ./hex2mif.perl
#   ./mipsel-readelf -a testvec > testvec.r
#   ./mipsel-nm testvec > testvec.n

    testvec.o: testvec.s
    $(AS) $(ASFLAGS) testvec.s -o testvec.o               ← Assembling (testvec.s)

    add.o:  add.c
    $(CPP) add.c > add.i                                  ← Preprocessing
    $(CC) -Wall -S add.i                                  ← Compiling the preprocessed file
    $(AS) $(ASFLAGS) add.s -o add.o                       ← Assembling (add.s)
#   $(CC) $(CCFLAGS) add.c
    $(OBJDUMP) -xS  add.o > add.dump                      ← Disassembling the object code
```

- Preprocessing

add.i

```
# 1 "add.c"
# 1 "<built-in>"
# 1 "<command line>"
# 1 "add.c"


int compare()
{
  int a, b, c;
  a = ((b) < (c) ? (b) : (c));;
  return a;
}

```

- Compilation

add.s

```
    .file   1 "add.c"
    .section .mdebug.abi32
    .previous
    .text
    .align  2
    .globl  compare
    .ent    compare
compare:
    .frame  $fp,32,$31      # vars= 24, regs= 1/0, args= 0, gp= 0
    .mask   0x40000000,-8
    .fmask  0x00000000,0
    .set    noreorder
    .set    nomacro

    addiu   $sp,$sp,-32
    sw  $fp,24($sp)
    move    $fp,$sp
    lw  $2,4($fp)
    nop
    sw  $2,20($fp)
    lw  $3,0($fp)
    nop
    sw  $3,16($fp)
    lw  $4,16($fp)
    lw  $3,20($fp)
    nop
    slt $2,$3,$4
    beq $2,$0,$L2
    nop

    lw  $4,20($fp)
    nop
    sw  $4,16($fp)
$L2:
    lw  $2,16($fp)
    nop
    sw  $2,8($fp)
    lw  $2,8($fp)
    move    $sp,$fp
    lw  $fp,24($sp)
    addiu   $sp,$sp,32
```

- Assembler (Disassembled result)

add.dump

```
Disassembly of section .text:

00000000 <compare>:
   0:   27bdffe0    addiu   sp,sp,-32
   4:   afbe0018    sw  s8,24(sp)
   8:   03a0f021    move    s8,sp
   c:   8fc20004    lw  v0,4(s8)
  10:   00000000    nop
  14:   afc20014    sw  v0,20(s8)
  18:   8fc30000    lw  v1,0(s8)
  1c:   00000000    nop
  20:   afc30010    sw  v1,16(s8)
  24:   8fc40010    lw  a0,16(s8)
  28:   8fc30014    lw  v1,20(s8)
  2c:   00000000    nop
  30:   0064102a    slt v0,v1,a0
  34:   10400004    beqz    v0,48 <compare+0x48>
  38:   00000000    nop
  3c:   8fc40014    lw  a0,20(s8)
  40:   00000000    nop
  44:   afc40010    sw  a0,16(s8)
  48:   8fc20010    lw  v0,16(s8)
  4c:   00000000    nop
  50:   afc20008    sw  v0,8(s8)
  54:   8fc20008    lw  v0,8(s8)
  58:   03c0e821    move    sp,s8
  5c:   8fbe0018    lw  s8,24(sp)
  60:   27bd0020    addiu   sp,sp,32
  64:   03e00008    jr  ra
  68:   00000000    nop
```

- Linker (Disassembled result)

testvec.dump

```
Disassembly of section .text:

00000000 <compare-0x10>:
   0:   0c000004    jal 10 <compare>
   4:   00000000    nop
   8:   ac020054    sw  v0,84(zero)
   c:   00000000    nop

00000010 <compare>:
  10:   27bdffe0    addiu   sp,sp,-32
  14:   afbe0018    sw  s8,24(sp)
  18:   03a0f021    move    s8,sp
  1c:   8fc20004    lw  v0,4(s8)
  20:   00000000    nop
  24:   afc20014    sw  v0,20(s8)
  28:   8fc30000    lw  v1,0(s8)
  2c:   00000000    nop
  30:   afc30010    sw  v1,16(s8)
  34:   8fc40010    lw  a0,16(s8)
  38:   8fc30014    lw  v1,20(s8)
  3c:   00000000    nop
  40:   0064102a    slt v0,v1,a0
  44:   10400004    beqz    v0,58 <compare+0x48>
  48:   00000000    nop
  4c:   8fc40014    lw  a0,20(s8)
```