ICS Homework Week 7

October 22, 2019

1 Operand Specifiers

Assume the following values are stored at the indicated memory addresses and registers:

Address	Value	Register	Value
0x100	0x78	%rax	0x100
0x104	0x56	%rcx	0x1
0x108	0x34	%rdx	0x3
0x10C	0x12		

Fill in the following table showing the values for the indicated operands:

Operand	Value
%rax	**0x100**
0x104	**0x56**
$0x108	**0x108**
(%rax)	**0x78**
4(%rax)	**0x56**
5(%rax,%rdx)	**0x34**
256(%rcx,%rdx)	**0x56**
0xFC(,%rdx,4)	**0x34**
(%rax,%rdx,4)	**0x12**

2 Data Movement

2.1

Assume variables *sp* and *dp* are declared with types

```
1 src_t *sp;
2 dst_t *dp;
```

where *src_t* and *dst_t* are data types declared with *typedef*. We wish to use the appropriate pair of data movement instructions to implement the operation

```
1 *dp = (dst_t) *sp;
```

Assume that the values of *sp* and *dp* are stored in registers *%rdi* and *%rsi*, respectively. For each entry in the table, show the two instructions that implement the specified data movement. The first instruction in the sequence should read from memory, do the appropriate conversion, and set the appropriate portion of register *%rax*. The second instruction should then write the appropriate portion of *%rax* to memory. In both cases, the portions may be *%rax*, *%eax*, *%ax*, or *%al*, and they may differ from one another.

Recall that when performing a cast that involves both a size change and a change of "signedness" in C, the operation should change the size first.

src_t	dst_t	Instruction
long	long	**movq (%rdi), %rax** **movq %rax, (%rsi)**
int	long	**movslq (%rdi), %rax** **movq %rax, (%rsi)**
char	unsigned	**movsbl (%rdi), %eax** **movl %eax, (%rsi)**
unsigned	long	**movl (%rdi), %eax** **movq %rax, (%rsi)**
int	char	**movl (%rdi), %eax** **movb %al, (%rsi)**
unsigned char	unsigned	**movzbl (%rdi), %eax** **movl %eax, (%rsi)**
char	short	**movsbw (%rdi), %ax** **movw %ax, (%rsi)**

2.2

You are given the following information. A function with prototype

```
1 void decode1(long *xp, long *yp, long *zp);
```

is compiled into assembly code, yield the following:

```
1   void decode1(long *xp, long *yp, long *zp)
2   xp in %rdi, yp in %rsi, zp in %rdx
3 decode1:
4   movq  (%rdi), %r8
5   movq  (%rsi), %rcx
6   movq  (%rdx), %rax
7   movq  %r8, (%rdx)
8   movq  %rcx, (%rdi)
9   movq  %rax, (%rsi)
```

Parameters *xp*, *yp*, and *zp* are stored in registers *%rdi*, *%rsi*, and *%rdx*, respectively.

Write C code for *decode*1 that will have an effect equivalent to the assembly code shown.

```
void decode1(long *xp, long *yp, long *zp) {
        long x = *xp;
        long y = *yp;
        long z = *zp;

        *zp = x;
        *xp = y;
        *yp = z;
}
```