**1.** Write a function that, given a number n, returns another number where the kth bit from the right is set to to 0.

Examples:

killKthBit(37, 3) = 33 because 3710 = 100**1**012 ~> 100**0**012 = 3310

killKthBit(37, 4) = 37 because the 4th bit is already 0.

**int killKthBit(int n, int k) {**

**return n & ~(1 << (k – 1))**

**}**

**2.** mov vs lea - describe the difference between the following:

movq (%rdx), %rax

leaq (%rdx), %rax

mov moves content pointed by the address in rdx to rax

lea moves address in rdx to rax

**3.** Invalid mov Instructions - Explain why these instructions would not be found in an assembly program.

|  |  |
| --- | --- |
| a) movl %eax, %rdx  Since movl corresponds to double-precision while eax is 32-bit, this instruction should replace movl with movq |  |
| b) movb %di, 8(%rdx)  This instruction moves 8-bit to 64-bit register, which should use movzbq or movsbq |  |
| c) movq (%rsi),8(%rbp)  Destination must be an address |  |
| d) movw 0xFF, (%eax) |  |

This instruction moves 16-bit to 32-bit, which should use movslw

**4.** What would be the corresponding instruction to move 64 bits of data from register %rax to register %rcx?

movq %rax, %rcx

**5.**

int cool1(int a, int b) {

if ( b < a )

return b;

else

return a;

}

int cool2(int a, int b) {

if ( a < b )

return a;

else

return b;

}

int cool3(int a, int b) {

unsigned ub = (unsigned) b;

if ( ub < a )

return a;

else

return ub;

}

**Which of the functions would compile into this assembly code:**

movl %esi, %eax # b at %eax

cmpl %eax, %edi # a – b

jge .L4 # if cmpl > 0, jump

movl %edi, %eax # %eax = a

.L4: ret # Return b

Cool 1

**6**. Operand Form Practice (see page 181 in textbook)

Assume the following values are stored in the indicated registers/memory addresses.

|  |  |  |  |
| --- | --- | --- | --- |
| Address | Value | Register | Value |
| 0x104 | 0x34 | %rax | 0x104 |
| 0x108 | 0xCC | %rcx | 0x5 |
| 0x10C | 0x19 | %rdx | 0x3 |
| 0x110 | 0x42 | %rbx | 0x4 |

Fill in the table for the indicated operands:

|  |  |  |  |
| --- | --- | --- | --- |
| Operand | Value | Operand | Value |
| $0x110 | 272 | 3(%rax, %rcx) | 0x19 |
| %rax | 0x104 | 256(, %rbx, 2) | 0xCC |
| 0x110 | 0x42 | (%rax, %rbx, 2) | 0x19 |
| (%rax) | 0x34 |  |  |
| 8(%rax) | 0x19 |  |  |
| (%rax, %rbx) | 0xCC |  |  |

**7.** Condition Codes and Jumps - Assume the addresses and registers are in the same state as in Problem 6. Does the following code result in a jump to .L2?

leaq (%rax, %rbx), %rdi

# leaq 0x108, rdi

cmpq $0x100, %rdi

# 0x108->0xCC – 0x100

jg .L2

No, because 0xCC is greater than $0x100