Imperial College London

TUTORIAL

IMPERIAL COLLEGE LONDON

DEPARTMENT OF COMPUTING

C113 Architecture

Lecturer:
Dr. Jana Giceva
j.giceva@imperial.ac.uk

Head Teaching Assistant:
Izaak Coleman
ic711@imperial.ac.uk

Date: February 12, 2018

1 Basic Assembly

1.1 Addressing modes

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

Address	Value
0x204	0xFF
0x208	0xCD
0x20C	0x21
0x210	0x11

Register	Value
%rax	0x2
%rcx	0x204
%rdx	0x3

Fill in the following table showing the types (*i.e.*, immediate, register, memory) and the values of the indicated operands:

Operand	Type	Memory Address	Value
%rax	Register	N/A	0x2
0x210	Abs. Memory	0x210	0x11
$0x210	Immediate	N/A	0x210
(%rcx)	Memory	0x204	0xFF
4(%rcx)	Memory	0x208	0xCD
5(%rcx, %rdx)	Memory	0x20C	0x21
519(%rdx, %rax)	Memory	0x20C	0x21
0x204(,%rax, 4)	Memory	0x20C	0x21
(%rcx,%rax, 2)	Memory	0x208	0xCD

1.2 Assembly instruction suffix

For each of the following lines of assembly language, determine the appropriate instruction suffix based on the operands. For example `mov` can be rewritten to `movb`, `movw`, or `movl`.

```
1: movl  %eax, (%esp)
2: movw  (%esp,%edx,4), %dx
3: movb  $0x21, %al
4: movb  (%eax), %dh
5: pushl $0xAB
6: movw  %dx, (%esp)
7: popl  %esi
```

1.3 Accessing information and Data movement

Each of the following lines of code generates an error message when we invoke the assembler. Explain what is wrong with each line.

```
1: movb $0xFC, (%bl)      -- Can’t use %bl as address register
2: movl %ax, (%esp)       -- Mismatch between instruction suffix and register ID
3: movw (%eax), 4(%esp)   -- Can’t have both src and dest be memory references
4: movb %ah, %sh          -- No register named %sh
5: movl %eax, $0x123      -- Can’t have immediate as destination
6: movl %eax,%dx          -- Destination operand has incorrect size
```