Dr Ryan Cunningham: room E142 (ryan.cunningham@mmu.ac.uk)

Manchester Metropolitan University

x64 Assembly Cheat Sheet (Linux)

x64 Registers

rax	(volatile) function return value
rbx	(non-volatile) general purpose
rcx	(volatile) 4th function parameter
rdx	(volatile) 3rd function parameter
rsi	(non-volatile) 2nd function parameter
rdi	(non-volatile) 1st function parameter
rbp	(non-volatile) stack base pointer
rsp	(non-volatile) stack top pointer
r8	(volatile) 5th function parameter
r9	(volatile) 6th function parameter
r10	(volatile) general purpose
r11	(volatile) general purpose
r12	(non-volatile) general purpose
r13	(non-volatile) general purpose
r14	(non-volatile) general purpose
r15	(non-volatile) general purpose

Branching/Jump Instructions

cmp %r11, %r10 jl label	jump to label if %r11 < %r10
cmp %r11, %r10 jg label	jump to label if %r11 > %r10
cmp %r11, %r10 jle label	jump to label if %r11 <= %r10
cmp %r11, %r10 jge label	jump to label if %r11 >= %r10
cmp %r11, %r10 je label	jump to label if %r11 == %r10
cmp %r11, %r10 jne label	jump to label if %r11 != %r10

Arithmetic Instructions

addq %r10, %r11	adds %r10 to %r11 and stores the result in %r11.
addq $1, %r11	adds 1 to %r11 and stores the result in %r11.
subq %r10, %r11	subtracts %r10 from %r11 and stores the result in %r11.
subq $1, %r11	subtracts 1 from %r11 and stores the result in %r11.
imul %r10, %r11	multiplies %r11 by %r10 and stores the result in %r11
imul $1, %r11	multiplies %r11 by 1 and stores the result in %r11

movq $0, rdx
movq $2, rbx
movq $11, rax
idiv %rbx
	divides %rax by %rbx, storing the quotient in %rax and the remainder in %rdx. Those specific registers must be used.

Example Program

```
.data
mystring: .asciz “Hello”
.global main
main:
    movq $mystring, %rdi
    subq $8, %rsp
    call printf
    addq $8, %rsp
    ret
```

Memory Access

movq -8(%rsp), %rax	retrieves the contents at memory location %rsp (the stack pointer) plus -8 and store in register %rbx
movq %rax, -8(%rsp)	move the contents %rax to memory location %rsp (the stack pointer) plus -8 and store in register %rbx
movq (%rsp, %rax, 8), %rbx	retrieves the contents of memory location at %rsp, plus %rax, multiplied by 8 and store in register %rbx
movq %rbx, (%rsp, %rax, 8)	move the contents of register %rax to memory location at %rsp, plus %rax, multiplied by 8

Online Assembler: https://www.onlinegdb.com/online_gcc_assembler

Book: Griffith, A., 2002. *GCC: the complete reference*. McGraw-Hill, Inc..