Info

x means destination register, SR means source register in a register–register operand, PR means register in a register–pointer operand. n means constant number (nB for 8-bit, nW for 16-bit, nD for 32-bit, nQ for 64-bit). * means pointer.
For arguments, R means register, K means constant, P means memory address.

Destination Registers (x)

`rax` ⇒ `0`
`rbx` ⇒ `3`
`rcx` ⇒ `1`
`rdx` ⇒ `2`
`rdi` ⇒ `7`

Source Registers (SR)

`R rax` ⇒ `Cx`
`R rbx` ⇒ `D(x+8)`
`R rcx` ⇒ `C(x+8)`
`R rdx` ⇒ `Dx`
`R rdi` ⇒ `F(x+8)`

Register in a Register–Pointer Op (PR)

`rax` → `04`
`rbx` → `1C`
`rcx` → `0C`
`rdx` → `14`
`rdi` → `3C`

1 mov

1.1 register

`mov R K` ⇒ `48 C7 Cx nQ`
`mov R R` ⇒ `48 89 SR`
`mov R P` ⇒ `48 8B PR 25 *`

1.2 pointer

`mov P R` ⇒ `48 89 PR 25 *`
`mov P K` ⇒ `48 C7 04 25 * nQ`

2 arithmetic

2.1 add

2.1.1 register

```
add R R ⇒ 48 01 SR
add R (K<0x80) ⇒ 48 83 Cx nB
add rax K ⇒ 48 05 nQ
add R K ⇒ 48 81 Cx nQ
add R P ⇒ 48 01 PR 25 *
```

2.1.2 pointer

```
add P R ⇒ 48 03 PR 25 *
```

2.2 sub

```
sub R S ⇒ 48 29 SR
sub R (K < 127) ⇒ 48 83 C(x+8) nB
sub rax K ⇒ 48 2D nQ
sub R K ⇒ 48 81 E(x+8) nQ
sub R P ⇒ 48 29 PR 25 *
```

2.2.1 pointer

```
sub P R ⇒ 48 2B PR 25 *
```

2.3 mul

```
mul R ⇒ 48 F7 Ex
mul P ⇒ 48 F7 24 25 *
```

2.4 imul

```
imul R ⇒ 48 F7 E(x+8)
imul P ⇒ 48 F7 2C 25 *
imul R R ⇒ 48 0F AF SR
```

2.5 div

```
div R ⇒ 48 F7 Fx
div P ⇒ 48 F7 34 25 *
```

2.6 idiv

```
idiv R ⇒ 48 F7 F(x+8)
idiv P ⇒ 48 F7 3C 25 *
```

2.7 neg

```
neg R ⇒ 48 F7 D(x+8)
neg P ⇒ 48 F7 1C 25 *
```

3 Shift

3.1 shr

```
shr R (K=1) ⇒ 48 D1 E(x+8)
shr R K ⇒ 48 C1 E(x+8) nB
shr P (K=1) ⇒ 48 D1 2C 25 *
shr P K ⇒ 48 C1 2C 25 * nB
```

3.2 sar

```
sar R (K=1) ⇒ 48 D1 F(x+8)
sar R K ⇒ 48 C1 F(x+8) nB
sar P (K=1) ⇒ 48 D1 3C 25 *
sar P K ⇒ 48 C1 3C 25 * nB
```

3.3 shl

```
shl R (K=1) ⇒ 48 D1 Ex
shl R K ⇒ 48 C1 Ex nB
shl R (K=1) ⇒ 48 D1 24 25 *
shl R K ⇒ 48 C1 24 25 * nB
```

3.4 ror

```
ror R (K=1) ⇒ 48 D1 C(x+8)
ror R K ⇒ 48 C1 C(x+8) nB
ror P (K=1) ⇒ 48 D1 0C 25 *
ror P K ⇒ 48 C1 0C 25 * nB
```

3.5 rol

```
rol R (K=1) ⇒ 48 D1 Cx
rol R K ⇒ 48 C1 Cx nB
rol P (K=1) ⇒ 48 D1 04 25 *
rol P K ⇒ 48 C1 04 25 * nB
```

4 Binary Logic

4.1 not

```
not R ⇒ 48 F7 Dx
not P ⇒ 48 F7 14 25 *
```

4.2 or

4.2.1 register

```
or R R ⇒ 48 09 SR
or R (K<0x80) ⇒ 48 83 C(x+8) nB
or rax K ⇒ 48 0D nQ
or R K ⇒ 48 81 C(x+8) nQ
or R P ⇒ 48 09 PR 25 *
```

4.2.2 pointer

```
or P R ⇒ 48 0B PR 25 *
or P (K<127) ⇒ 48 83 0C 25 * nB
or P K ⇒ 48 81 0C 25 * nQ
```

4.3 xor

```
xor R R ⇒ 48 31 SR
xor R P ⇒
```

4.4 and

```
and R R ⇒ 48 21 SR
```

5 Miscellaneous

```
nop ⇒ 90
syscall ⇒ 0F 05
```