# Assembly Light

Simple knowledge dump

HW

# Code Evaluation

## Read bytes of memory

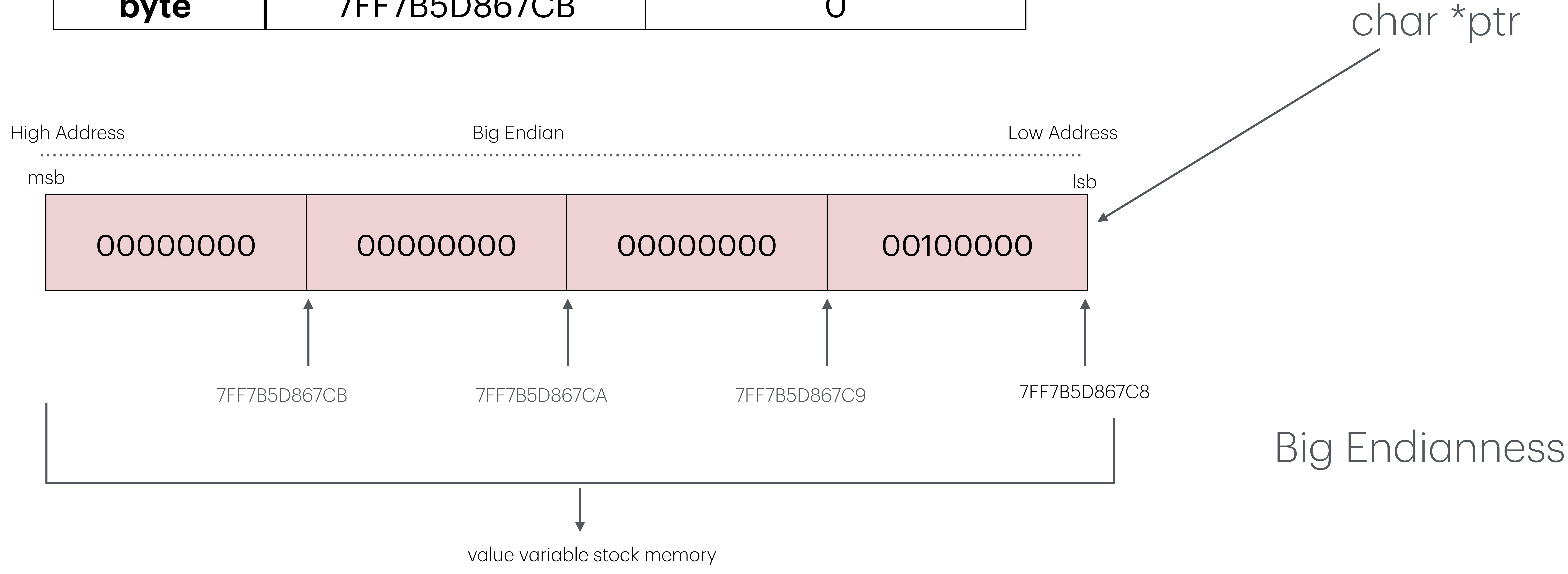
```
#include <iostream>

int main()
{
  int value;
  char *ptr =(char *) &value;

  value = 32;

  std::cout << "byte address: " << (const void  *) (ptr) << " " <<  std::endl;
  std::cout << "byte data: " <<  (int)*ptr << " " <<  std::endl;

  std::cout << "byte address + 1: " << (const void *) (ptr + 1)  << " "   << std::endl;
  std::cout << "byte data: " <<  (int)*(ptr+1) << " " <<  std::endl;

  std::cout << "byte address + 2: " <<  (const void *) (ptr + 2)  << " " << std::endl;
  std::cout << "byte data: " <<  (int)*(ptr+2) << " " <<  std::endl;

  std::cout << "byte address + 3: " <<  (const void *) (ptr + 3) << " " << std::endl;
  std::cout << "byte data: " <<  (int)*(ptr+3) << " " <<  std::endl;

  return 0;
}
```

# Code Evaluation

## Read bytes of memory

int value = 32

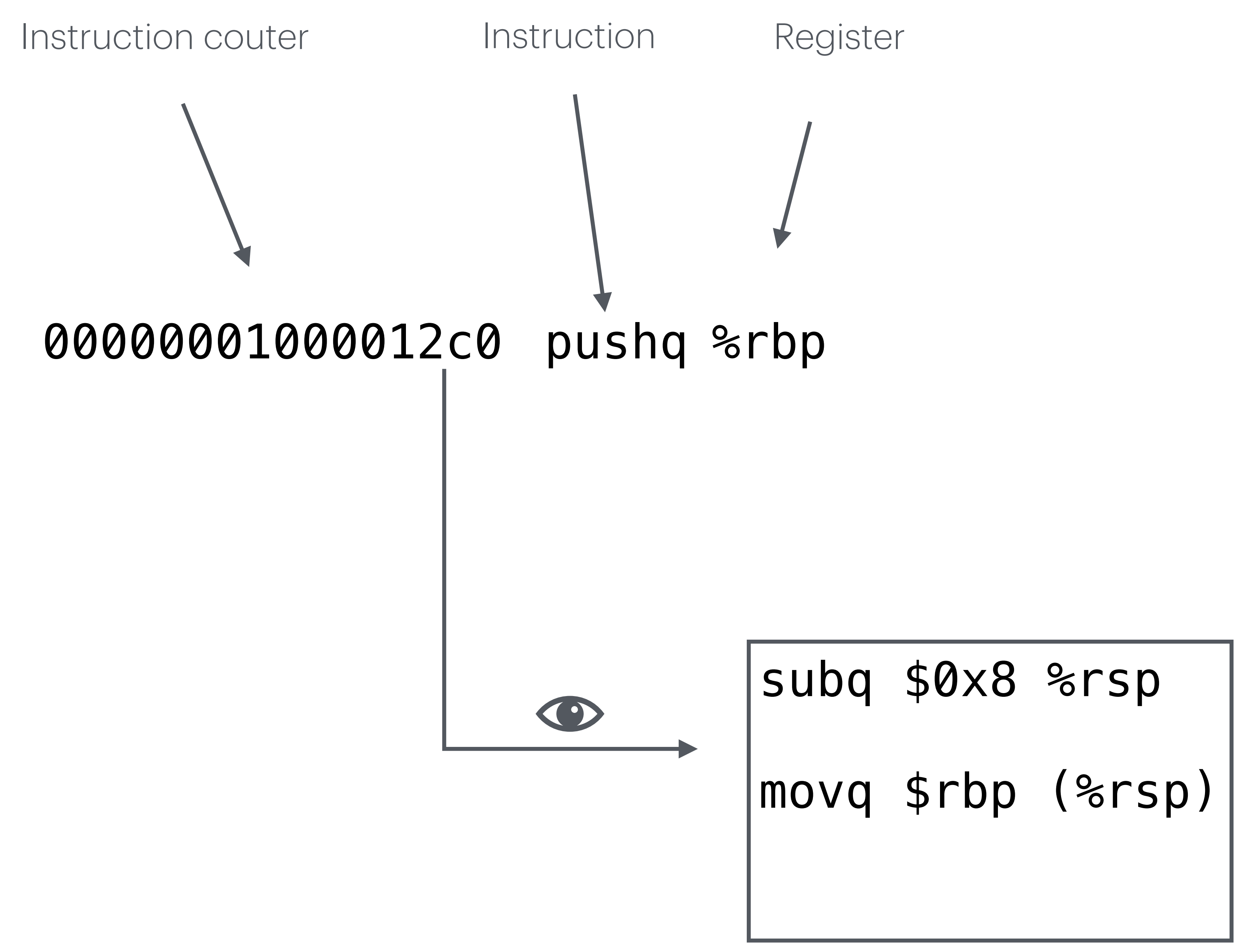
| | Byte Address (Hex) | Data (int) |
|---|---|---|
| **byte** | 7FF7B5D867C8 | 32 |
| **byte** | 7FF7B5D867C9 | 0 |
| **byte** | 7FF7B5D867CA | 0 |
| **byte** | 7FF7B5D867CB | 0 |



Big Endianness :)

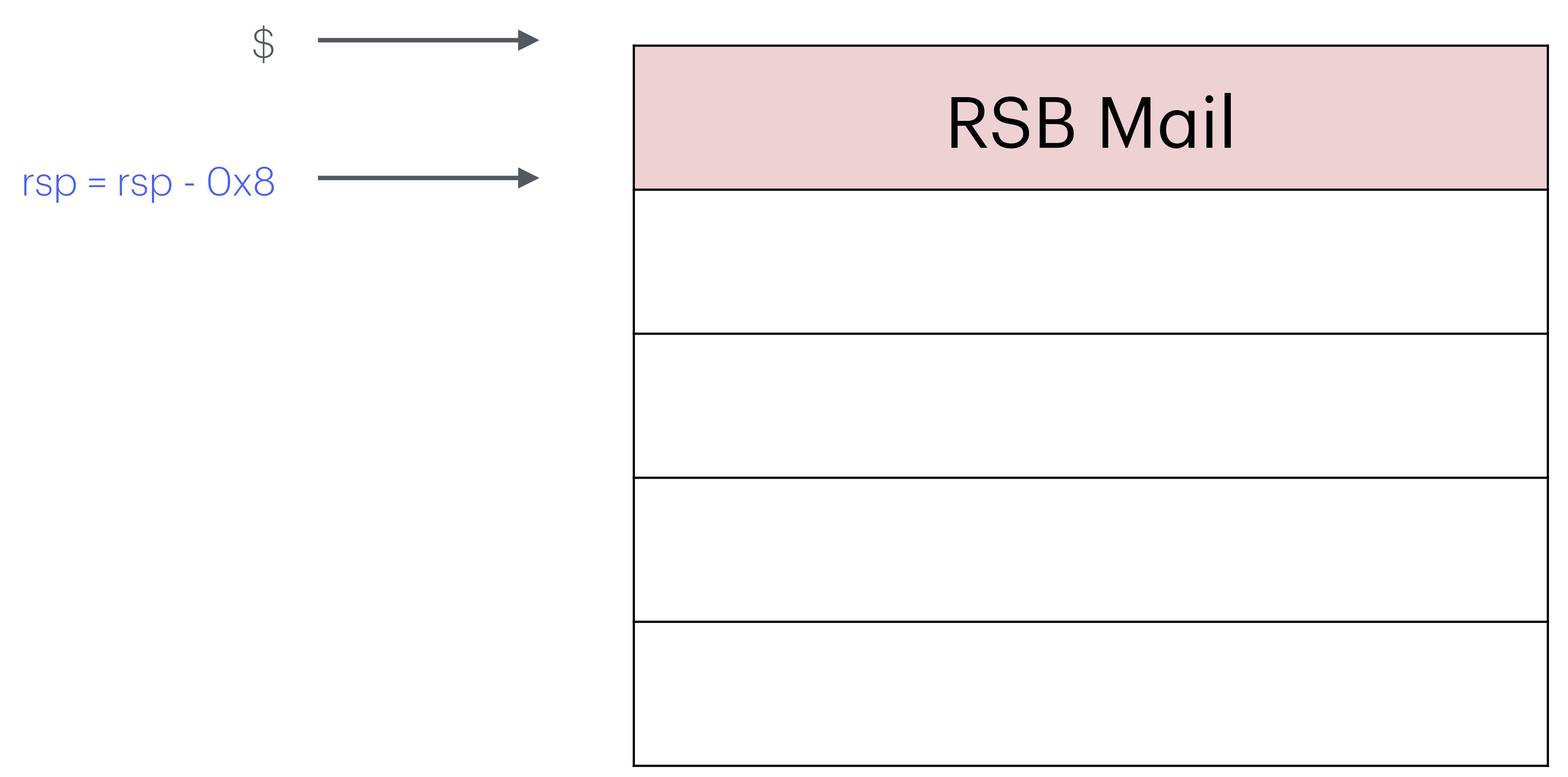
```cpp
#include <iostream>

int main()
{
  std::cout << "hello world.";
  return 0;
}
```

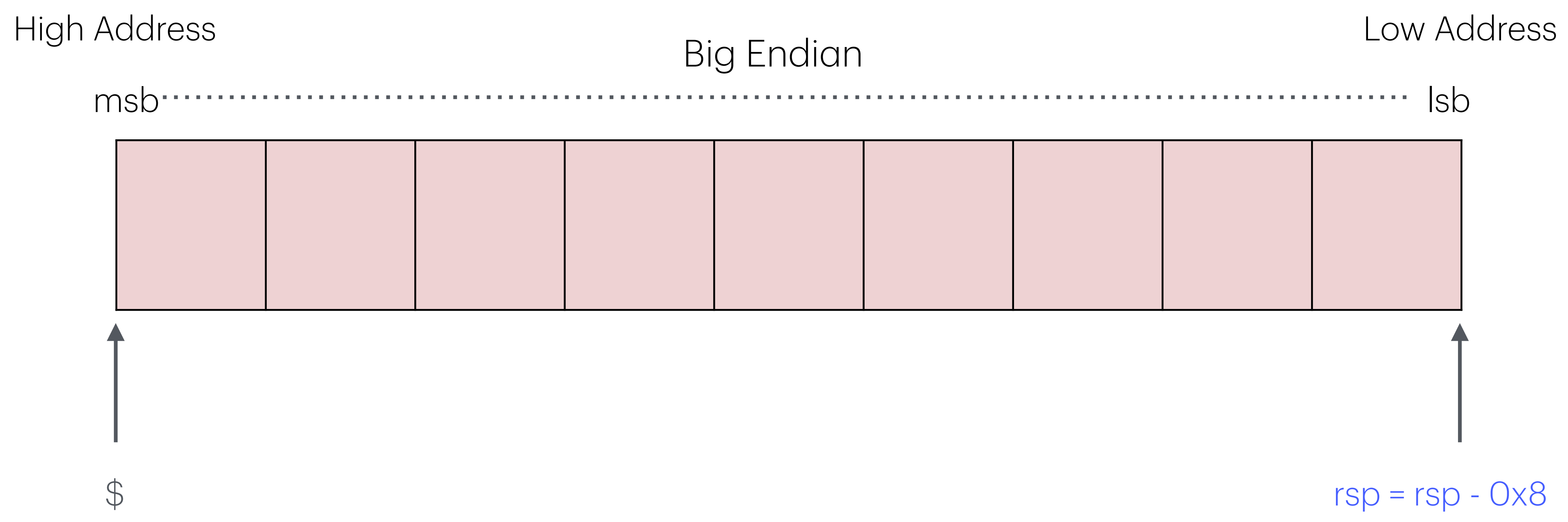
```
hello.o:
(__TEXT,__text) section
_main:
00000001000012c0 pushq %rbp
00000001000012c1 movq  %rsp, %rbp
00000001000012c4 subq  $0x10, %rsp
00000001000012c8 movl  $0x0, -0x4(%rbp)
00000001000012cf movq  0xd62(%rip), %rdi              ## literal pool symbol address: __ZNSt3__14coutE
00000001000012d6 leaq  0xc67(%rip), %rsi              ## literal pool for: "hello world."
00000001000012dd callq __ZNSt3__1lsB8ne180100INS_11char_traitsIcEEEERNS_13basic_ostreamIcT_EES6_PKc ##
std::__1::basic_ostream<char, std::__1::char_traits<char>>&
std::__1::operator<<[abi:ne180100]<std::__1::char_traits<char>>(std::__1::basic_ostream<char, std::__1::char_traits<char>>&, char
const*)
00000001000012e2 xorl  %eax, %eax
00000001000012e4 addq  $0x10, %rsp
00000001000012e8 popq  %rbp
00000001000012e9 retq
00000001000012ea nopw  (%rax,%rax)
```



```
00000001000012c0  pushq %rbp
```

```
subq $0x8 %rsp

movq $rbp (%rsp)
```

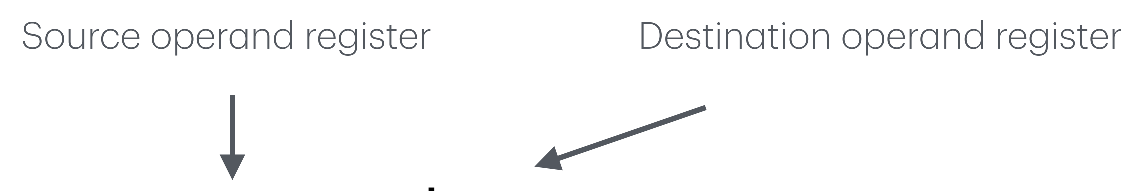


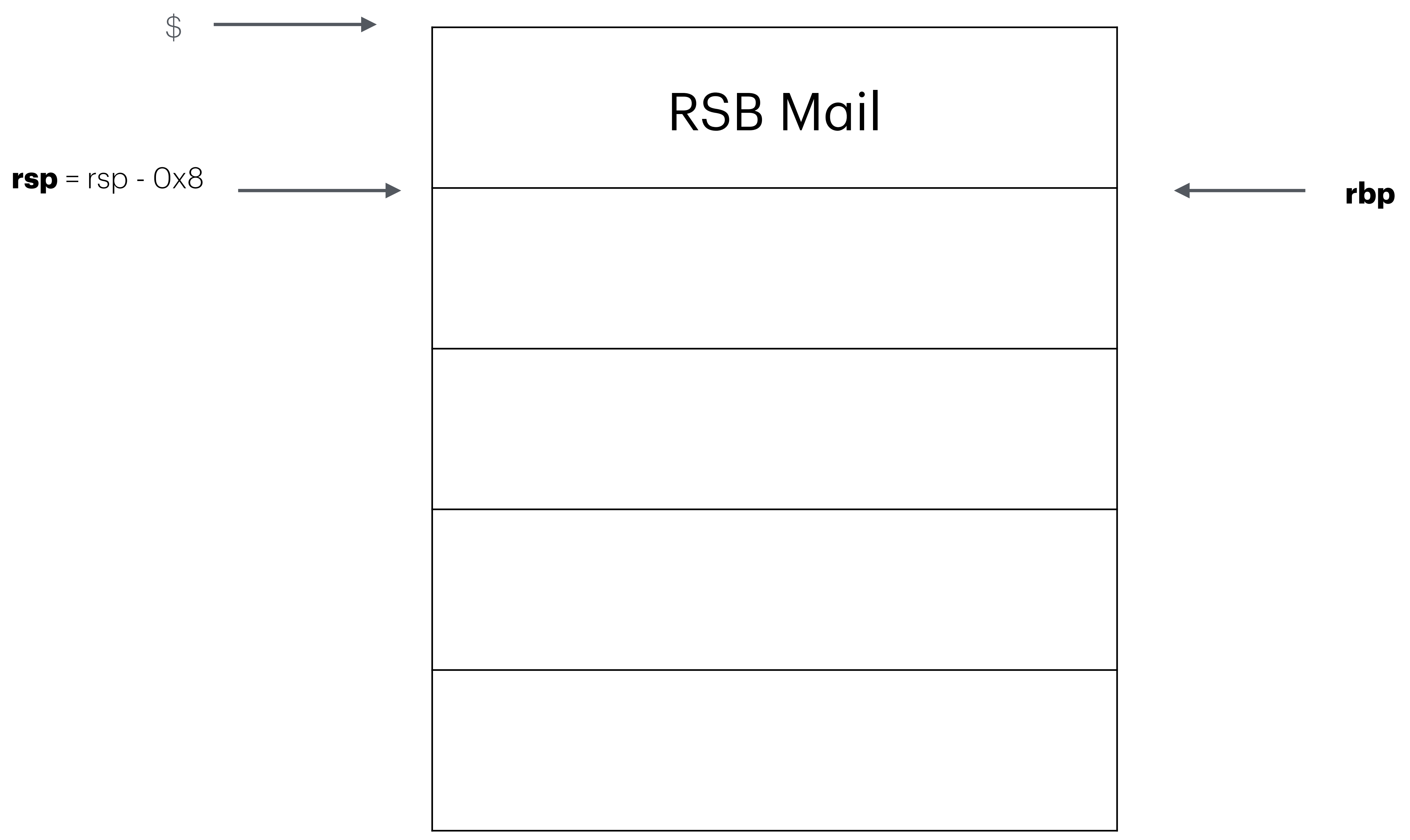


1) subtract 0x8 from register rsp

2) Copy register rbp into stack memory address rsp

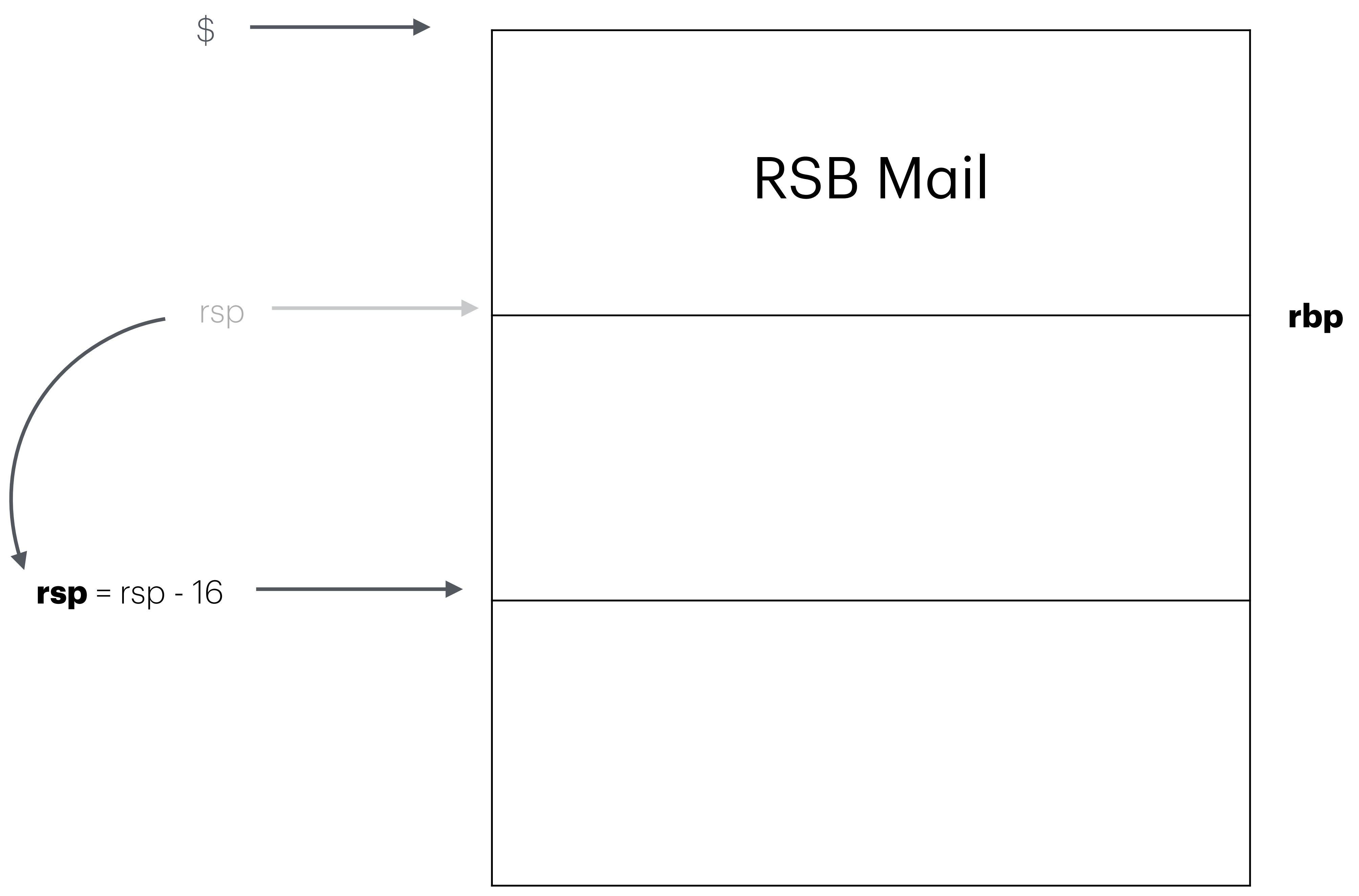
```
00000001000012c1  movq  %rsp, %rbp
```





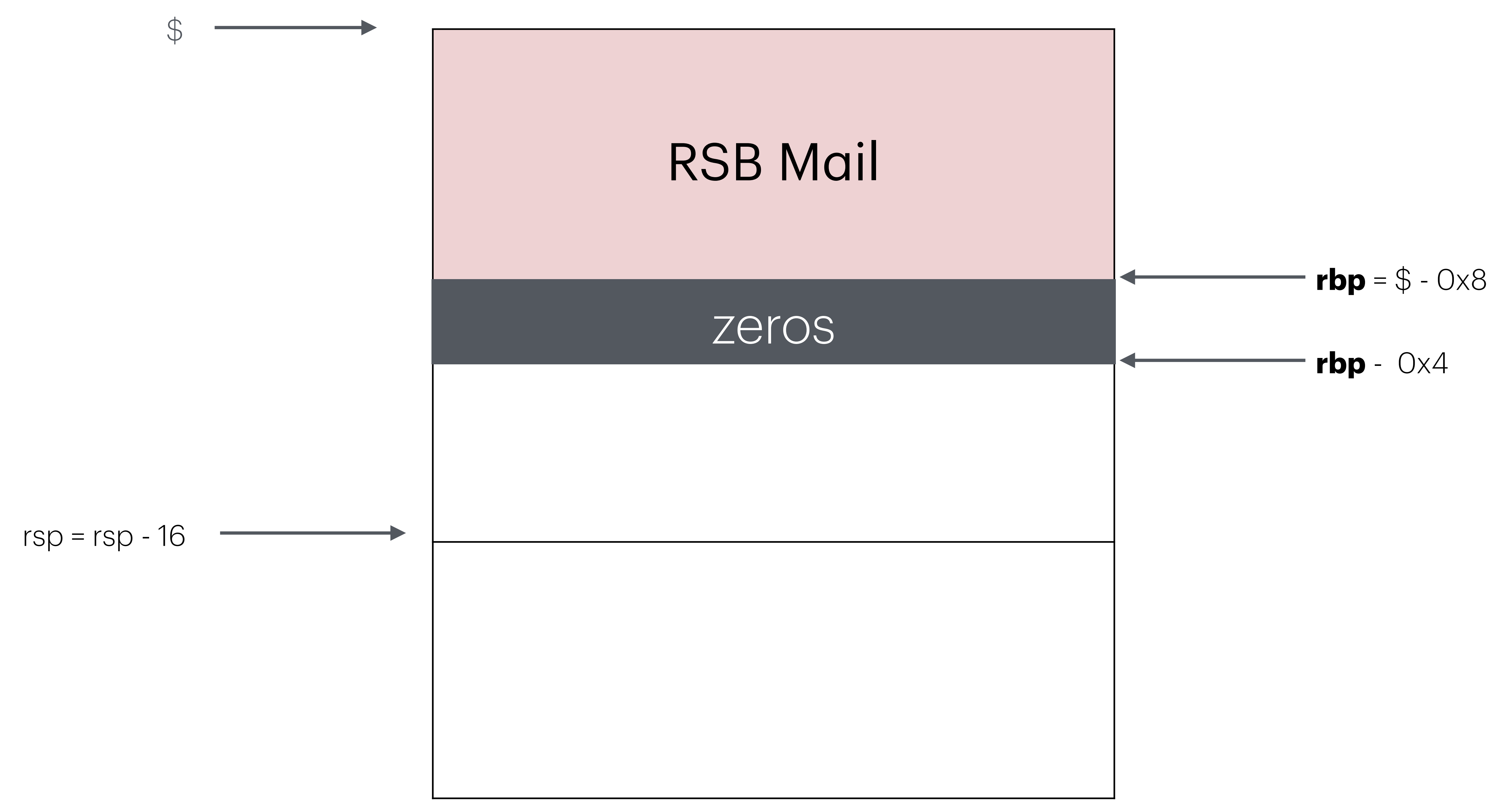
Copy register rsp into register rbp

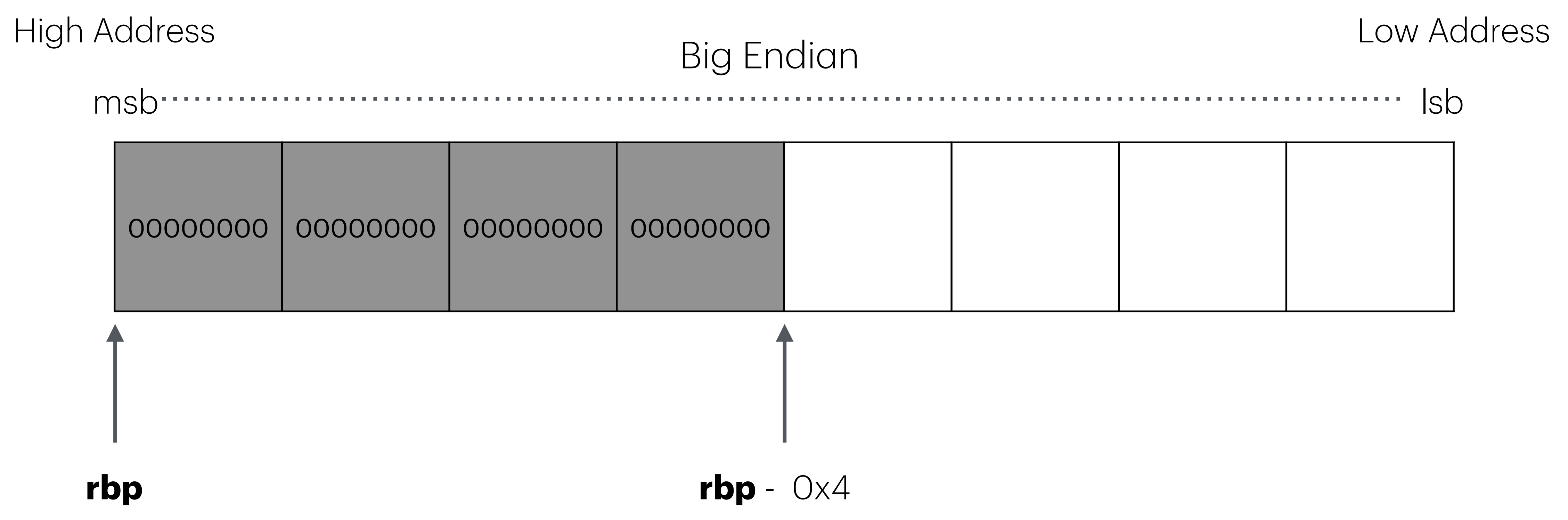
```
00000001000012c4  subq  $0x10, %rsp
```



Source immediate
Destination memory address

```
00000001000012c8  movl  $0x0, -0x4(%rbp)
```



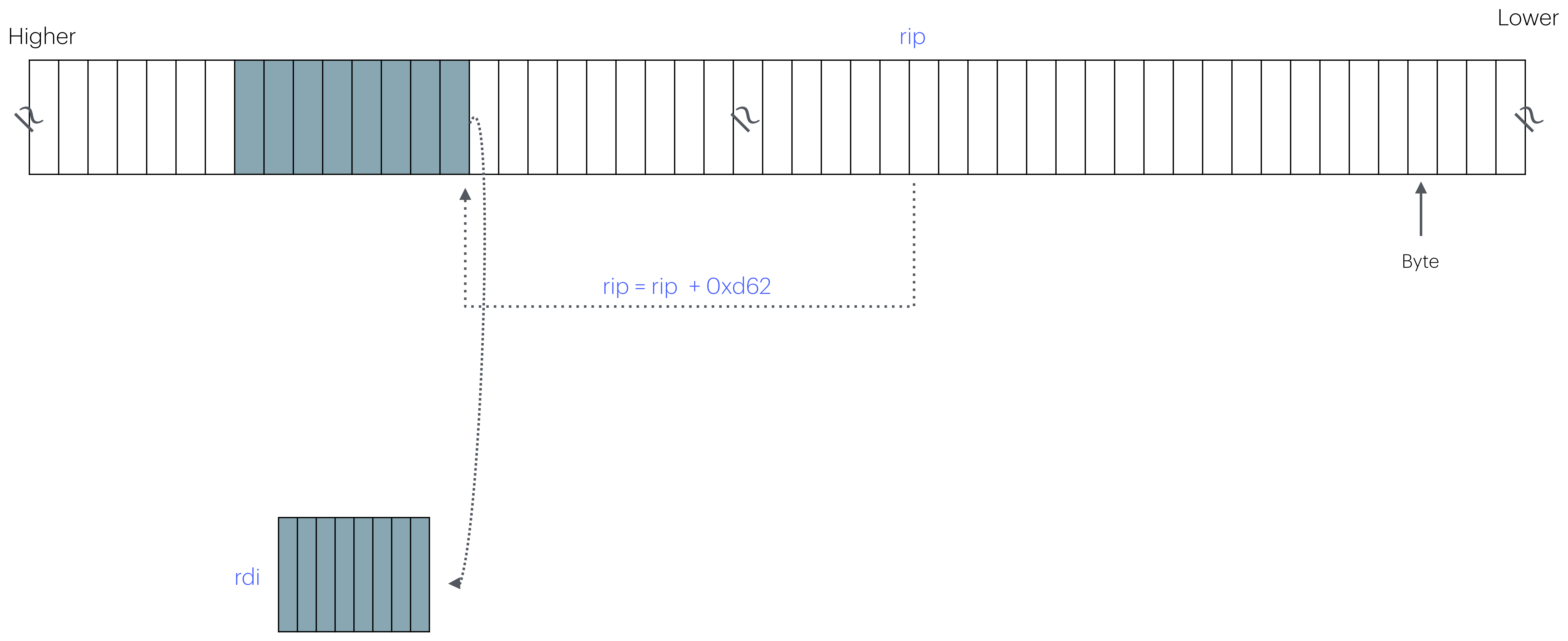


Load 4 byte zeros in stack destination address

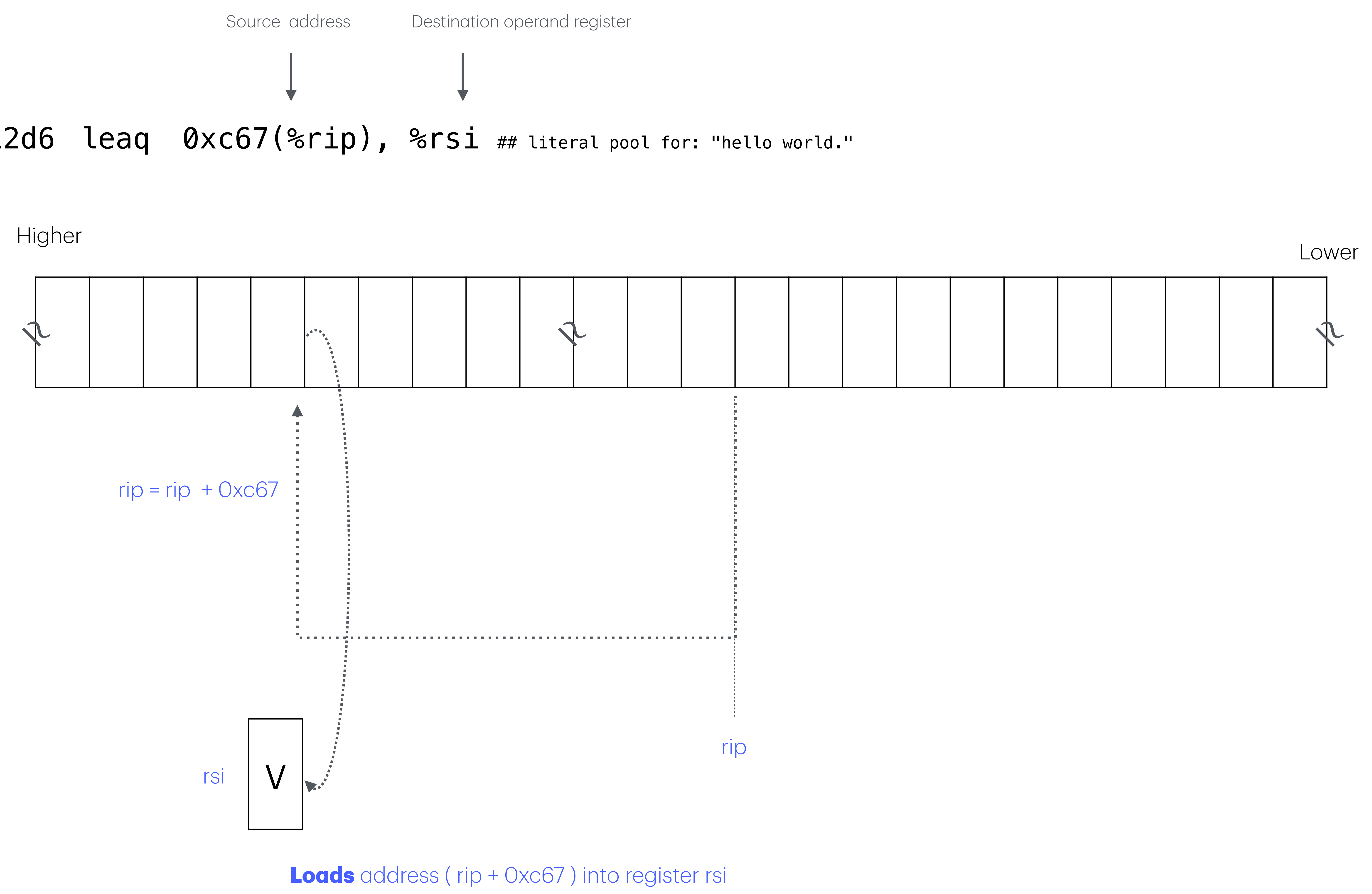
Source operand memory address

Destination operand register

```
00000001000012cf  movq  0xd62(%rip), %rdi ## literal pool symbol address: __ZNSt3__14coutE
```



**Transfer** 8 bytes from memory location Δrip into register rdi



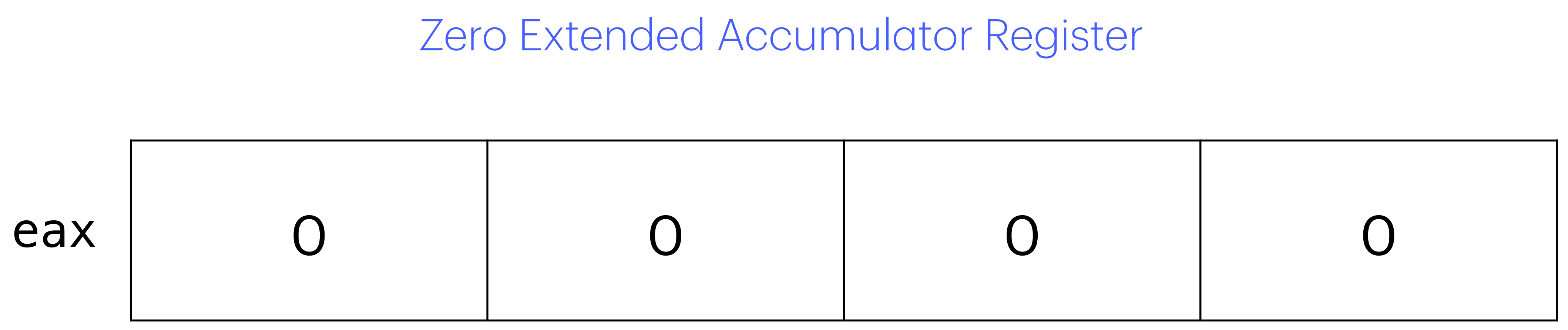
```
00000001000012d6  leaq  0xc67(%rip), %rsi ## literal pool for: "hello world."
```

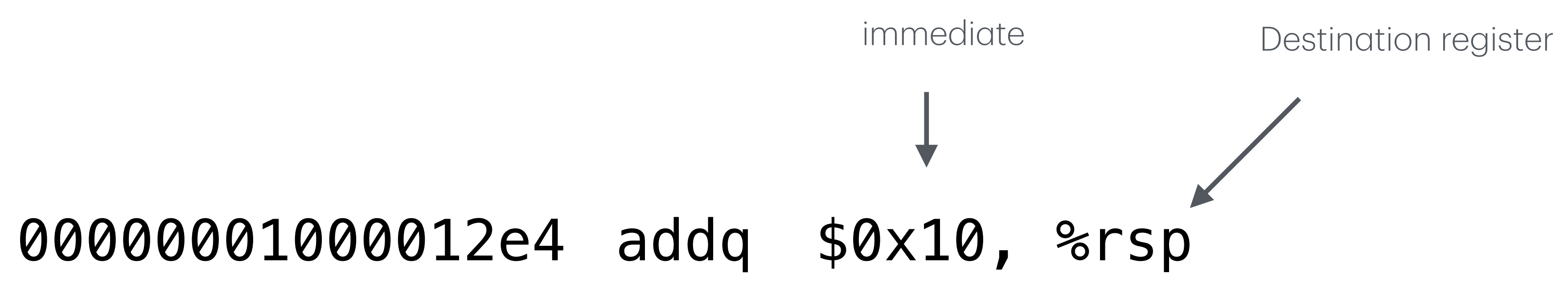
**Loads** address ( rip + 0xc67 ) into register rsi

```
00000001000012dd  callq
__ZNSt3__1lsB8ne180100INS_11char_traitsIcEEEERNS_13basic_ostreamIcT_EES6_PKc ##
std::__1::basic_ostream<char, std::__1::char_traits<char>>&
std::__1::operator<<[abi:ne180100]<std::__1::char_traits<char>>(std::__1::basic_ostream<char,
std::__1::char_traits<char>>&, char const*)
```

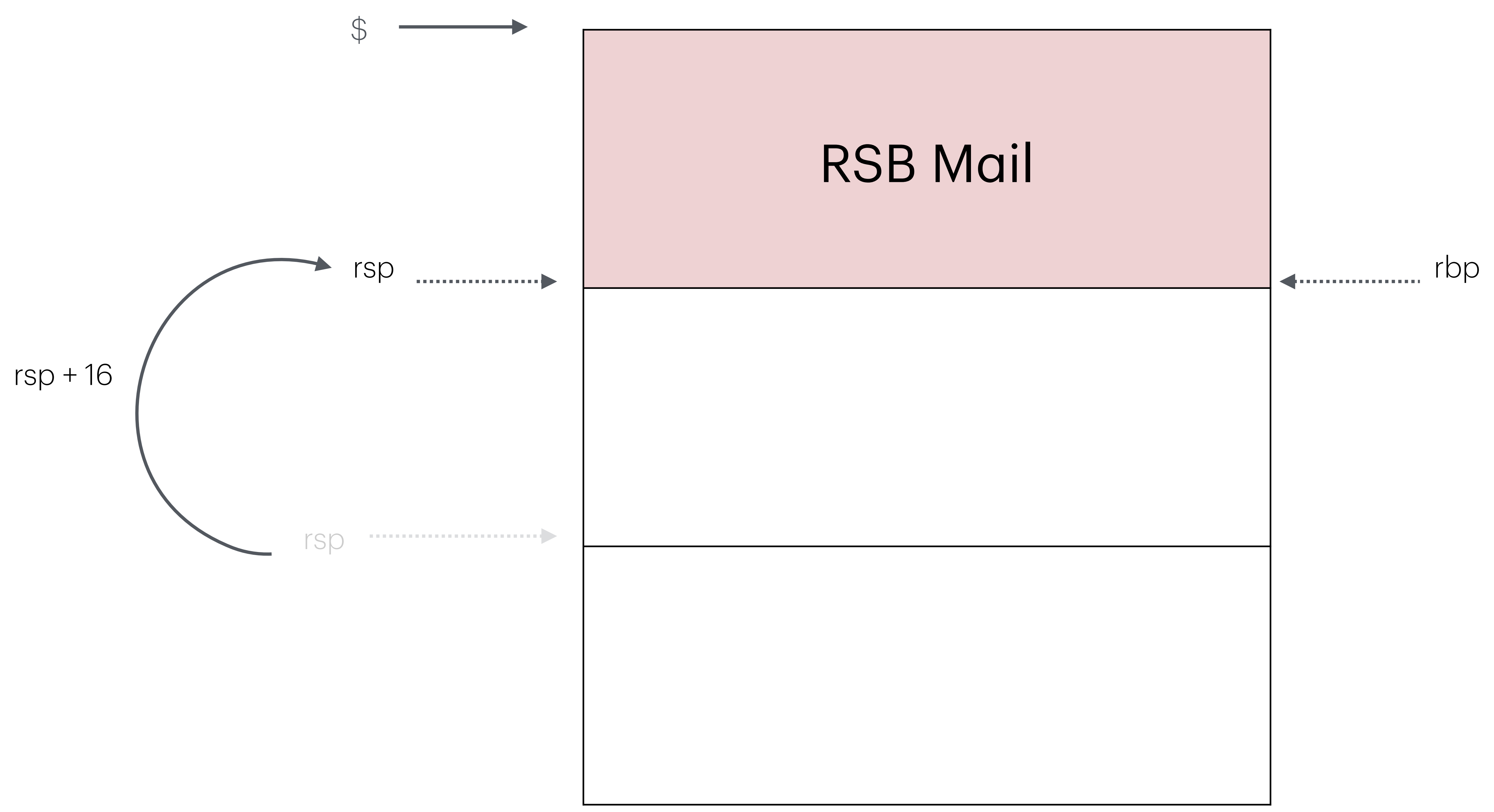
Call Routine

```
00000001000012e2  xorl  %eax, %eax
```





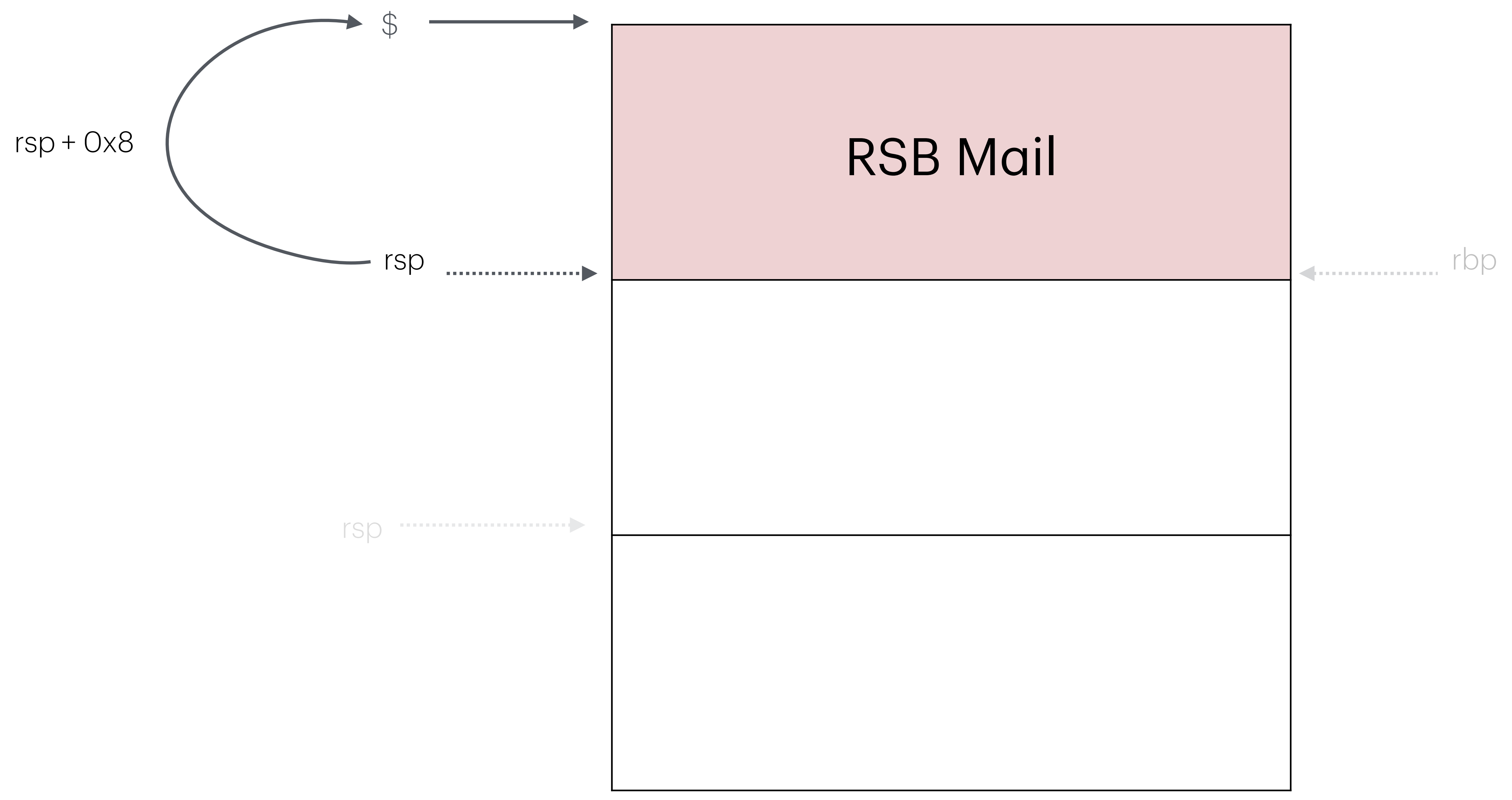
```
00000001000012e4  addq  $0x10, %rsp
```



Move back up the stack ( + 16 bytes )

```
0000000100012e8  popq  %rbp
```

```
movq ($rsp)%rbp

addq $0x8 %rsp
```

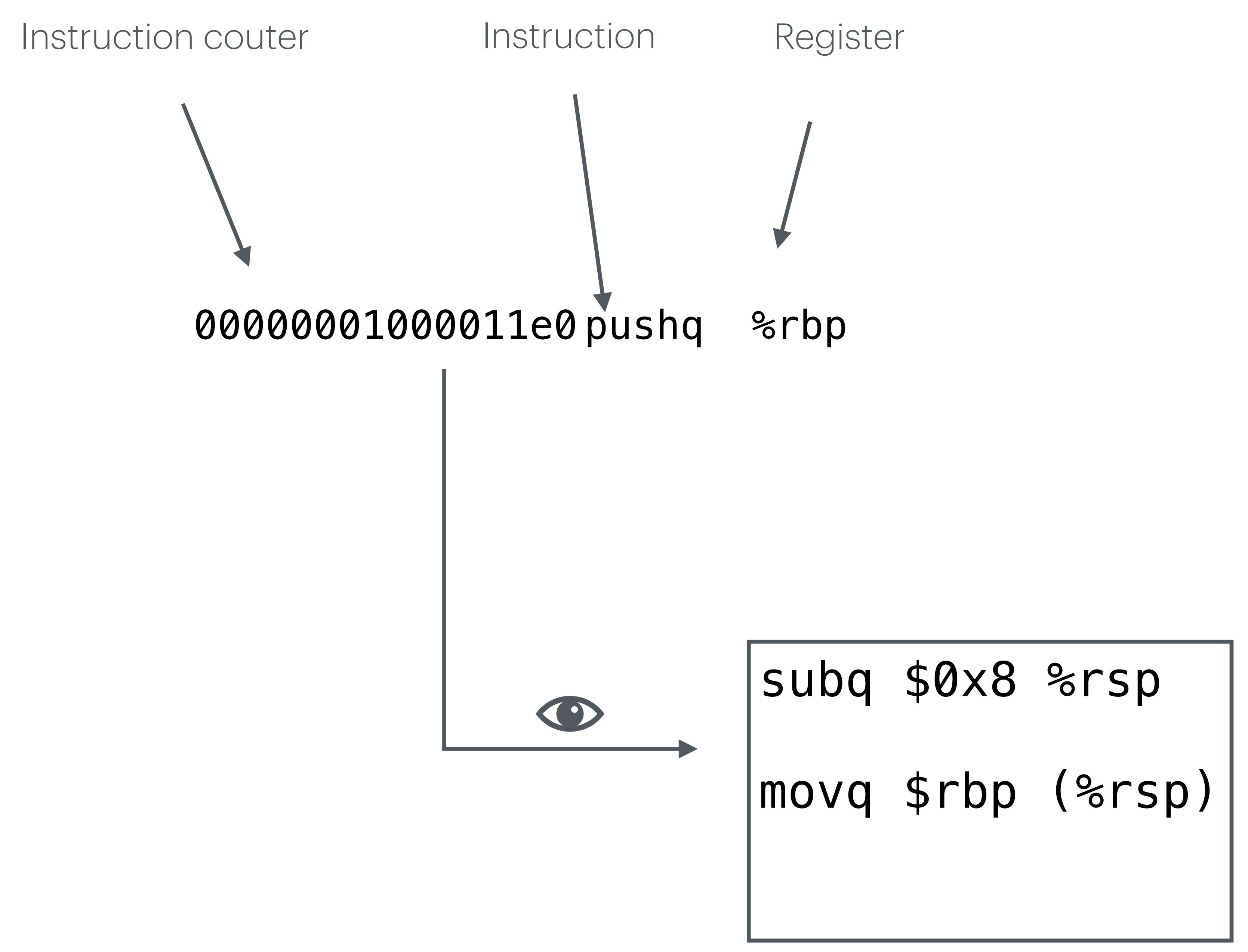


Copies 8 bytes from memory address in register rsp into register rbp

Move stack pointer back to stack head

```
#include <iostream>

int main()
{
  int value;
  int *ptr = &value;

  value = 32;

  std::cout << value  << std::endl;

  std::cout <<  ptr  << std::endl;


  std::cout << "hello world.";
  return 0;
}
```

```
hello2:
(__TEXT,__text) section
_main:
00000001000011e0 pushq  %rbp
00000001000011e1 movq %rsp, %rbp
00000001000011e4 subq $0x10, %rsp
00000001000011e8 movl $0x0, -0x4(%rbp)
00000001000011ef leaq -0x8(%rbp), %rax
00000001000011f3 movq %rax, -0x10(%rbp)
00000001000011f7 movl $0x20, -0x8(%rbp)
00000001000011fe movl -0x8(%rbp), %esi
0000000100001201 movq 0xe50(%rip), %rdi               ## literal pool symbol address: __ZNSt3__14coutE
0000000100001208 callq  0x100001e58                     ## symbol stub for: __ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsEi
000000010000120d movq %rax, %rdi
0000000100001210 leaq __ZNSt3__14endlB8ne180100IcNS_11char_traitsIcEEEERNS_13basic_ostreamIT_T0_EES7_(%rip), %rsi ## std::__1::basic_ostream<char,
std::__1::char_traits<char>>& std::__1::endl[abi:ne180100]<char, std::__1::char_traits<char>>(std::__1::basic_ostream<char, std::__1::char_traits<char>>&)
0000000100001217 callq  __ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsB8ne180100EPFRS3_S4_E ## std::__1::basic_ostream<char,
std::__1::char_traits<char>>::operator<<[abi:ne180100](std::__1::basic_ostream<char, std::__1::char_traits<char>>& (*)(std::__1::basic_ostream<char,
std::__1::char_traits<char>>&))
000000010000121c movq -0x10(%rbp), %rsi
0000000100001220 movq 0xe31(%rip), %rdi               ## literal pool symbol address: __ZNSt3__14coutE
0000000100001227 callq  0x100001e52                     ## symbol stub for: __ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsEPKv
000000010000122c movq %rax, %rdi
000000010000122f leaq __ZNSt3__14endlB8ne180100IcNS_11char_traitsIcEEEERNS_13basic_ostreamIT_T0_EES7_(%rip), %rsi ## std::__1::basic_ostream<char,
std::__1::char_traits<char>>& std::__1::endl[abi:ne180100]<char, std::__1::char_traits<char>>(std::__1::basic_ostream<char, std::__1::char_traits<char>>&)
0000000100001236 callq  __ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsB8ne180100EPFRS3_S4_E ## std::__1::basic_ostream<char,
std::__1::char_traits<char>>::operator<<[abi:ne180100](std::__1::basic_ostream<char, std::__1::char_traits<char>>& (*)(std::__1::basic_ostream<char,
std::__1::char_traits<char>>&))
000000010000123b movq 0xe16(%rip), %rdi               ## literal pool symbol address: __ZNSt3__14coutE
0000000100001242 leaq 0xcf3(%rip), %rsi               ## literal pool for: "hello world."
0000000100001249 callq  __ZNSt3__1lsB8ne180100INS_11char_traitsIcEEEERNS_13basic_ostreamIcT_EES6_PKc ## std::__1::basic_ostream<char, std::__1::char_traits<char>>&
std::__1::operator<<[abi:ne180100]<std::__1::char_traits<char>>(std::__1::basic_ostream<char, std::__1::char_traits<char>>&, char const*)
000000010000124e xorl %eax, %eax
0000000100001250 addq $0x10, %rsp
0000000100001254 popq %rbp
0000000100001255 retq
0000000100001256 nopw %cs:(%rax,%rax)
```

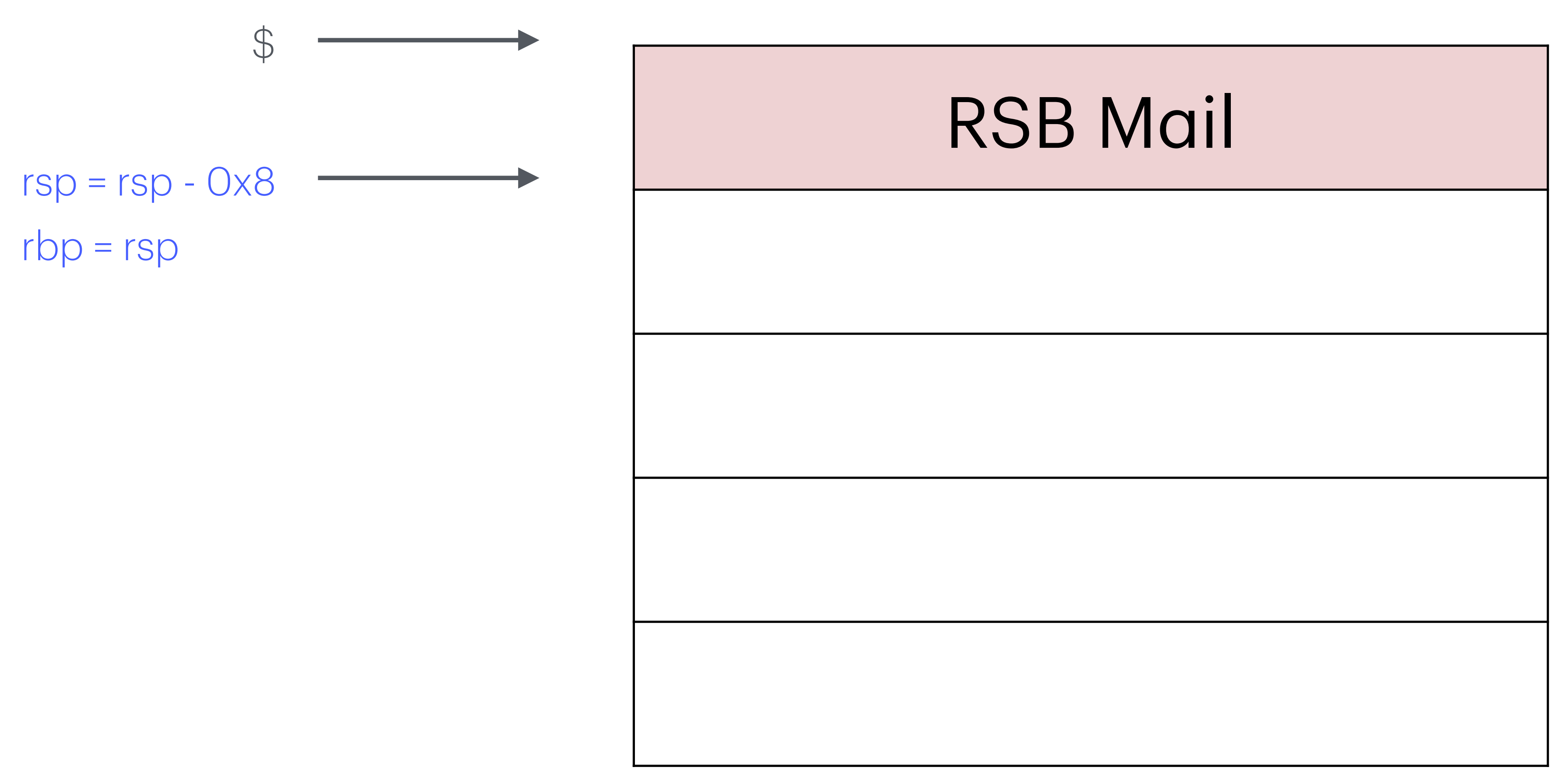


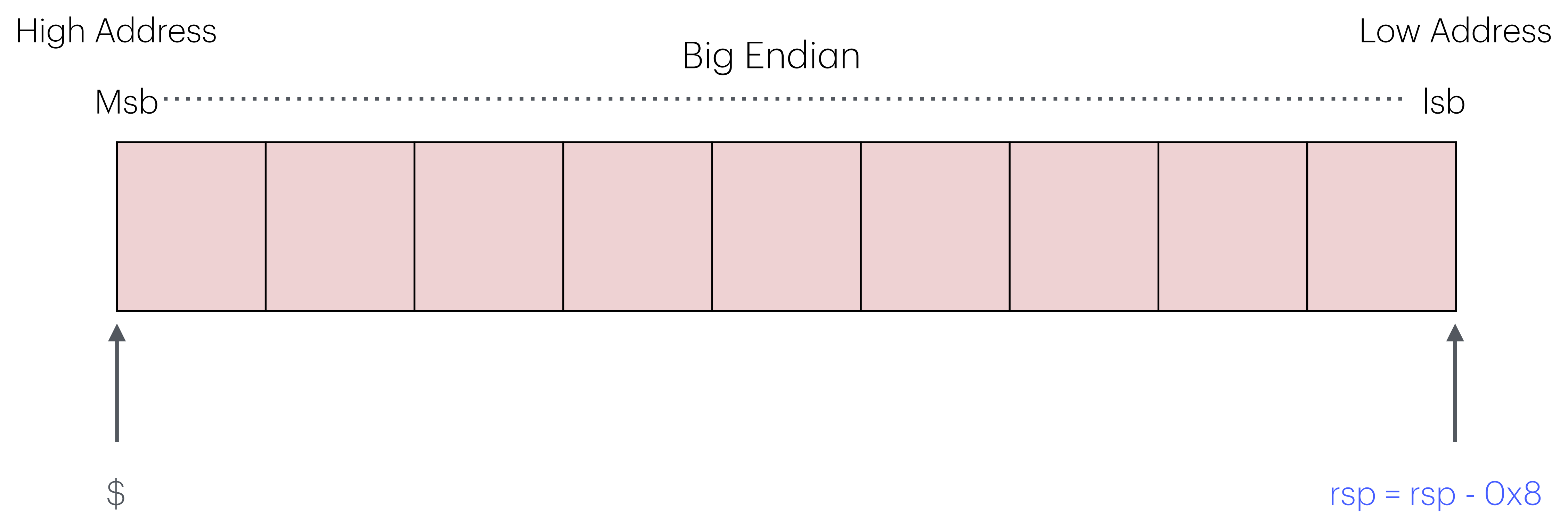
```
00000001000011e0 pushq  %rbp
```

```
subq $0x8 %rsp

movq $rbp (%rsp)
```

subtract 0x8 from register rsp

Copy value of rbp into rsp memory address ( refer to RSB mail in above image $ )

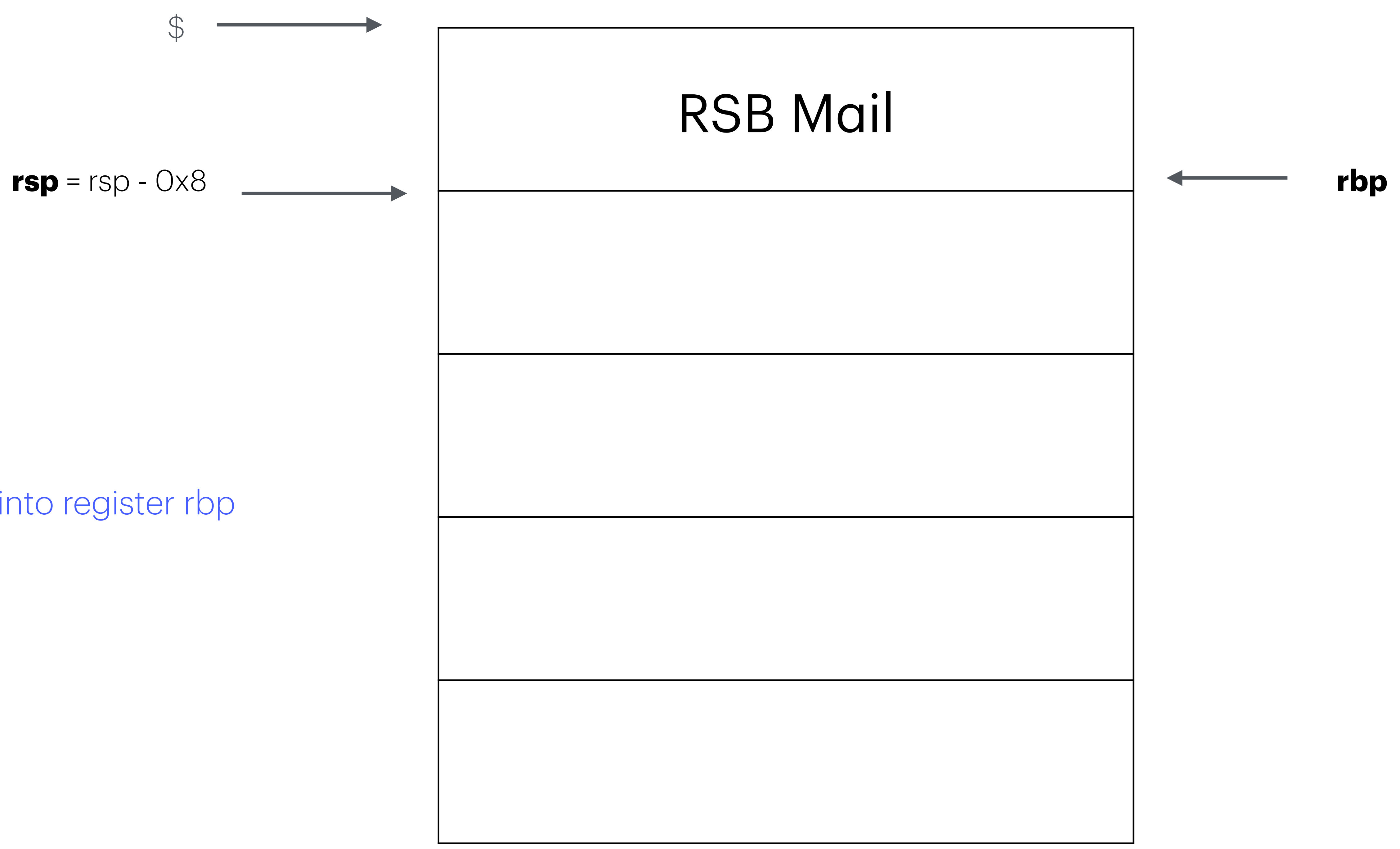




Source operand register

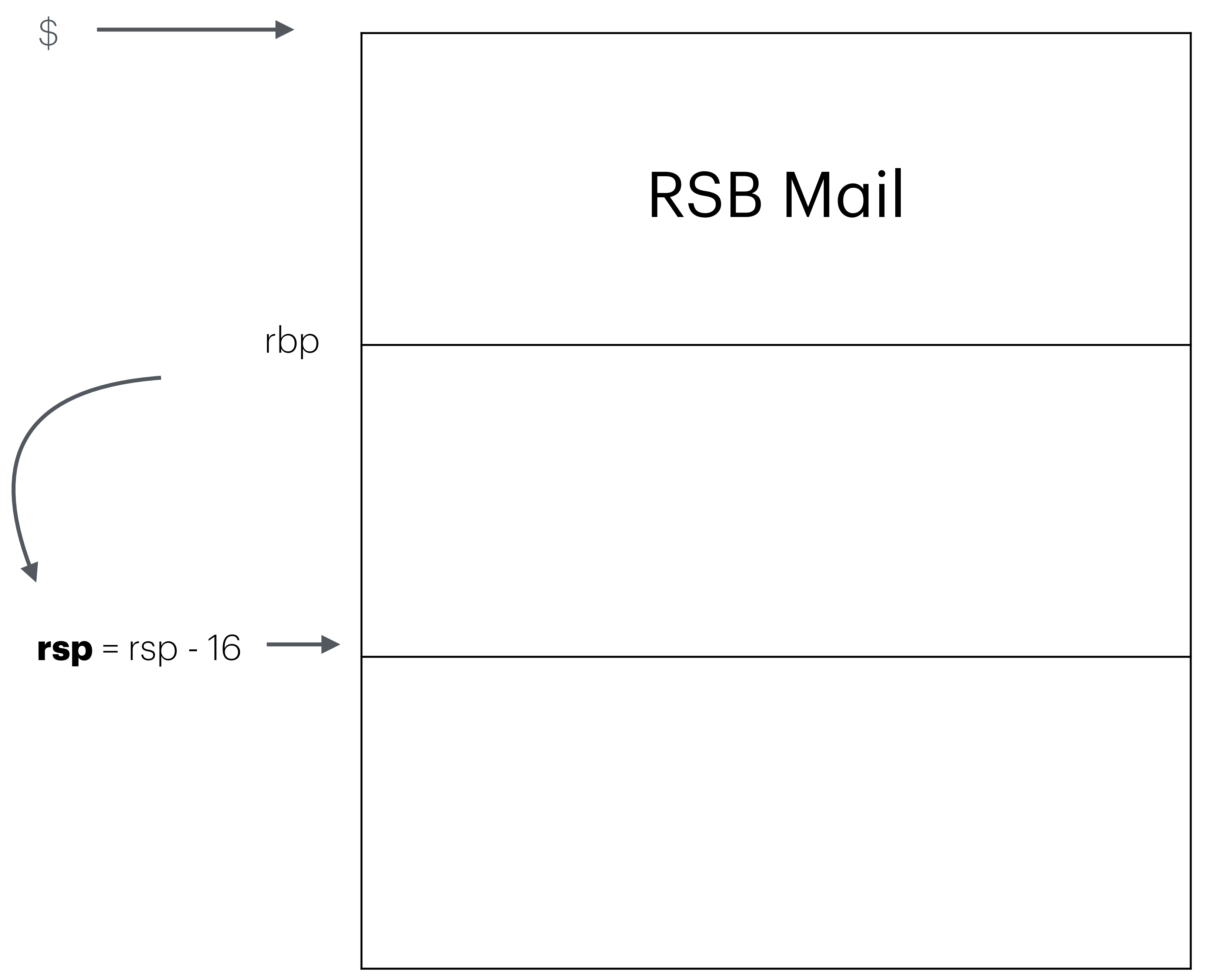
Destination operand register

```
00000001000011e1 movq %rsp, %rbp
```

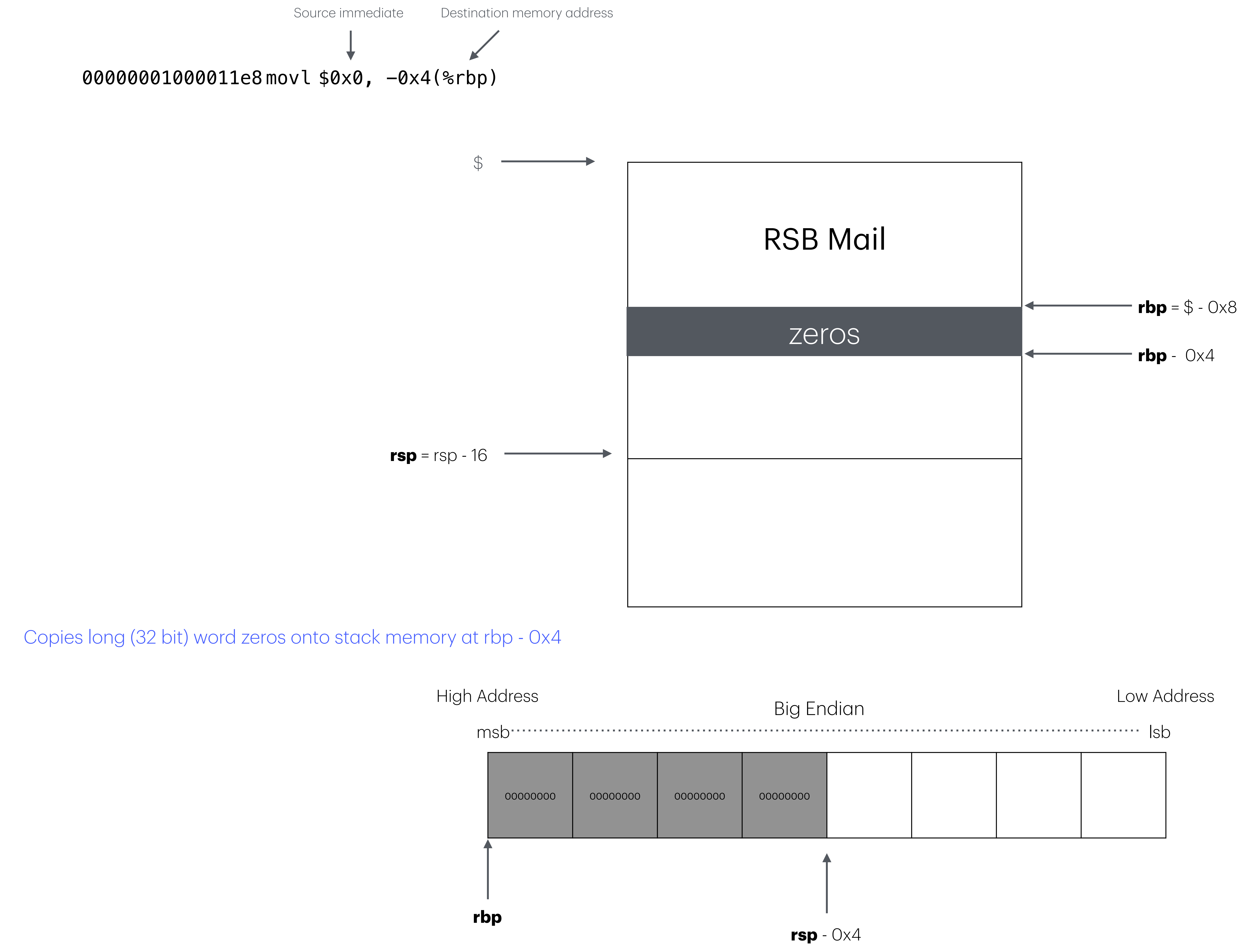


Copy value of register rsp into register rbp

```
00000001000011e4 subq  $0x10, %rsp
```



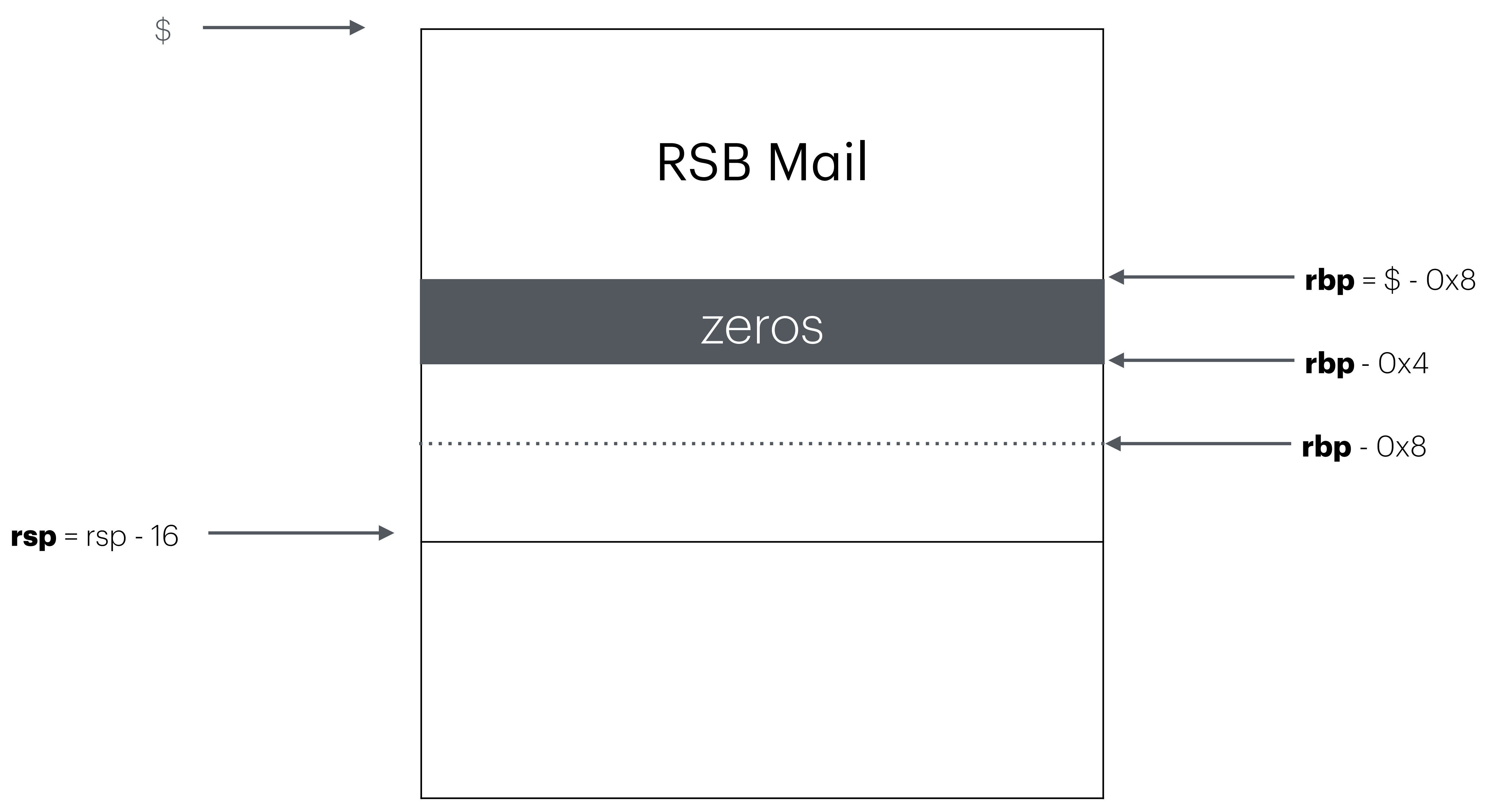
```
00000001000011e8 movl $0x0, -0x4(%rbp)
```



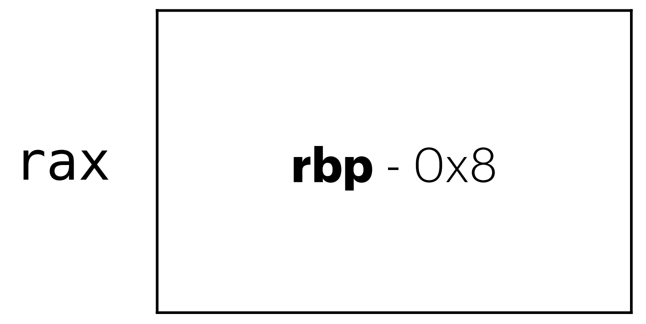
Copies long (32 bit) word zeros onto stack memory at rbp - 0x4

```
00000001000011ef leaq -0x8(%rbp), %rax
```

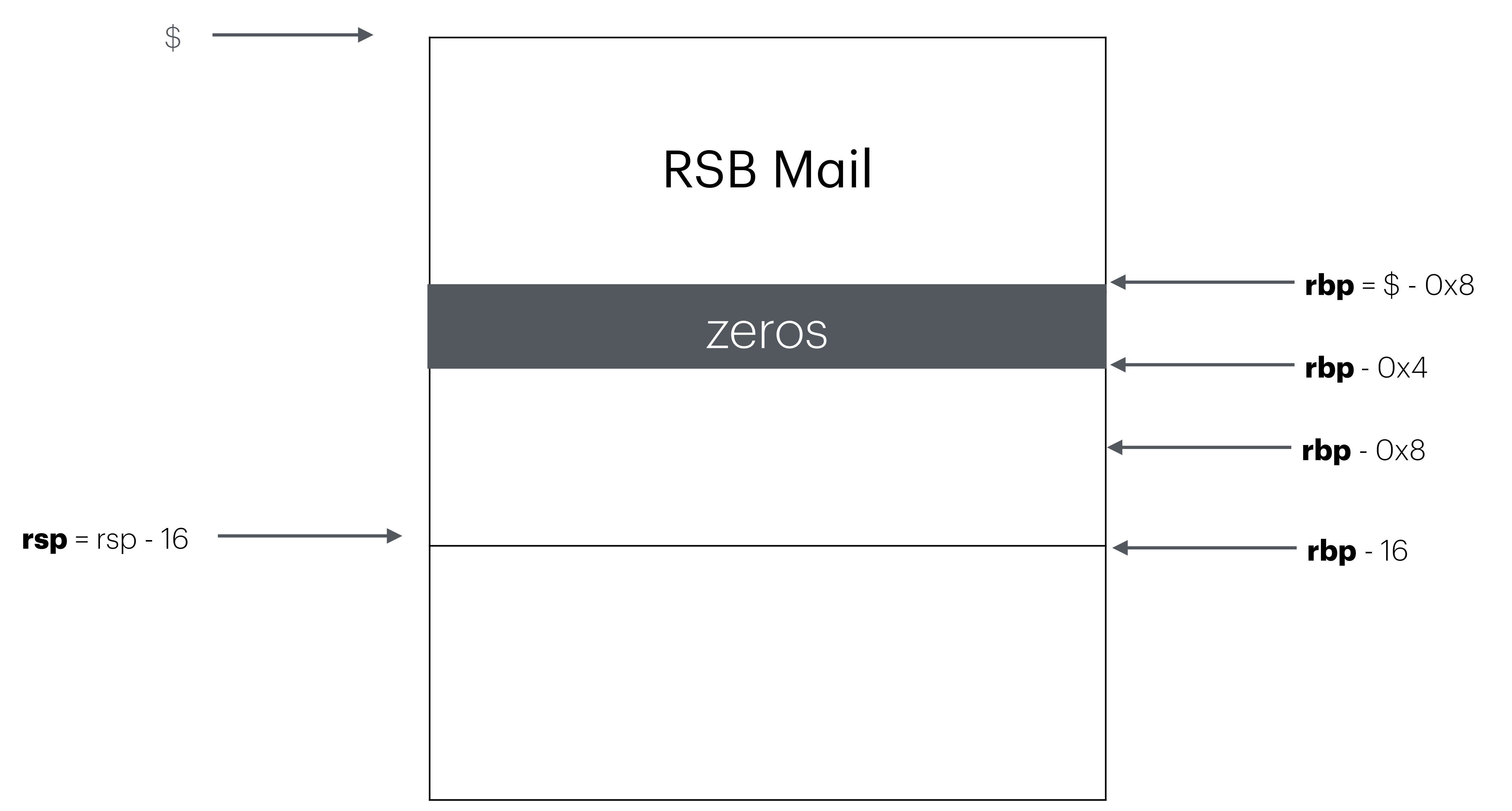
```
rax - 64 bit general purpose register
```

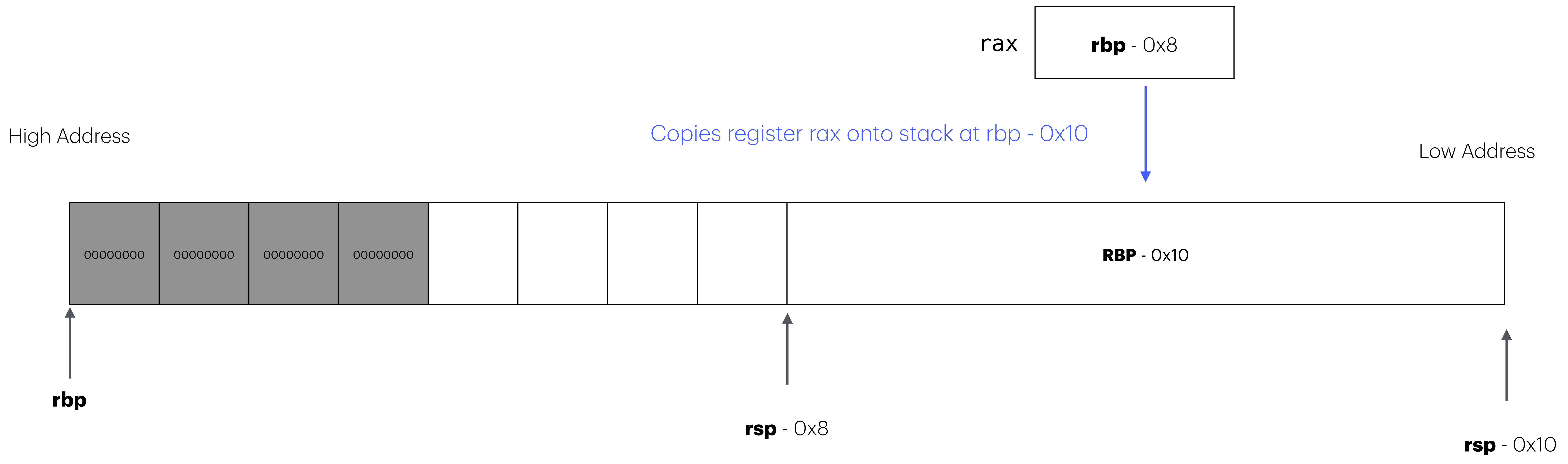


Load stack address rbp - 0x8 into rax register

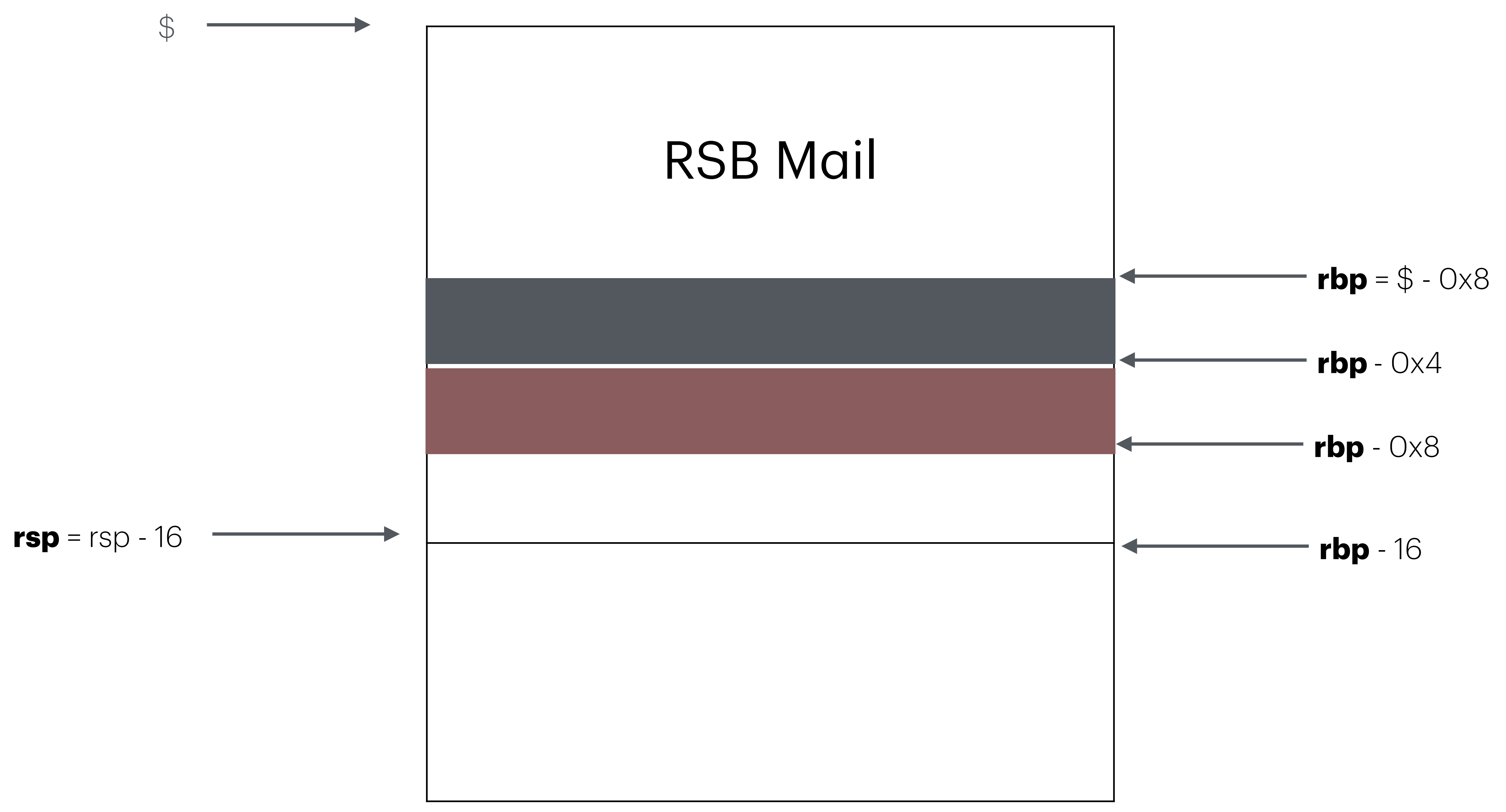


```
00000001000011f3 movq %rax, −0x10(%rbp)
```

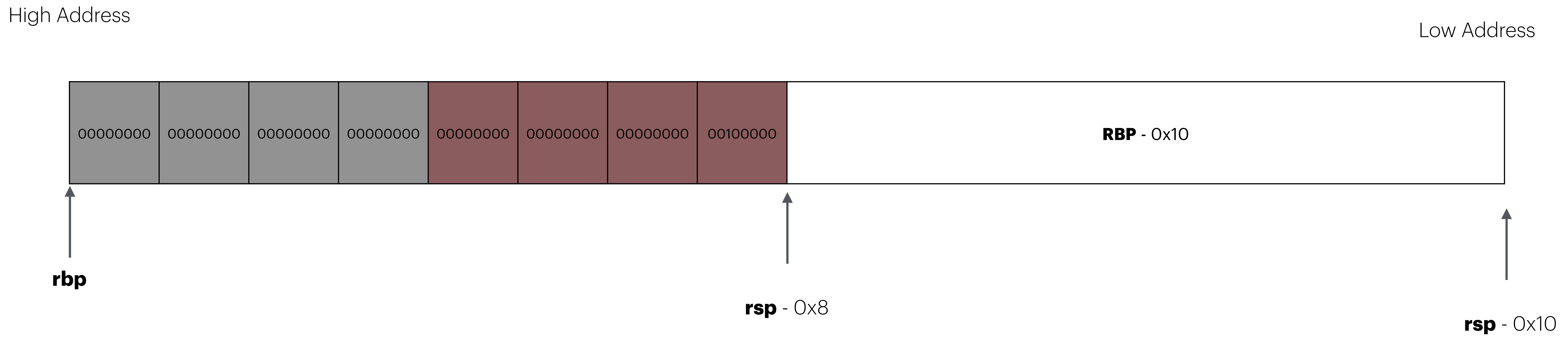




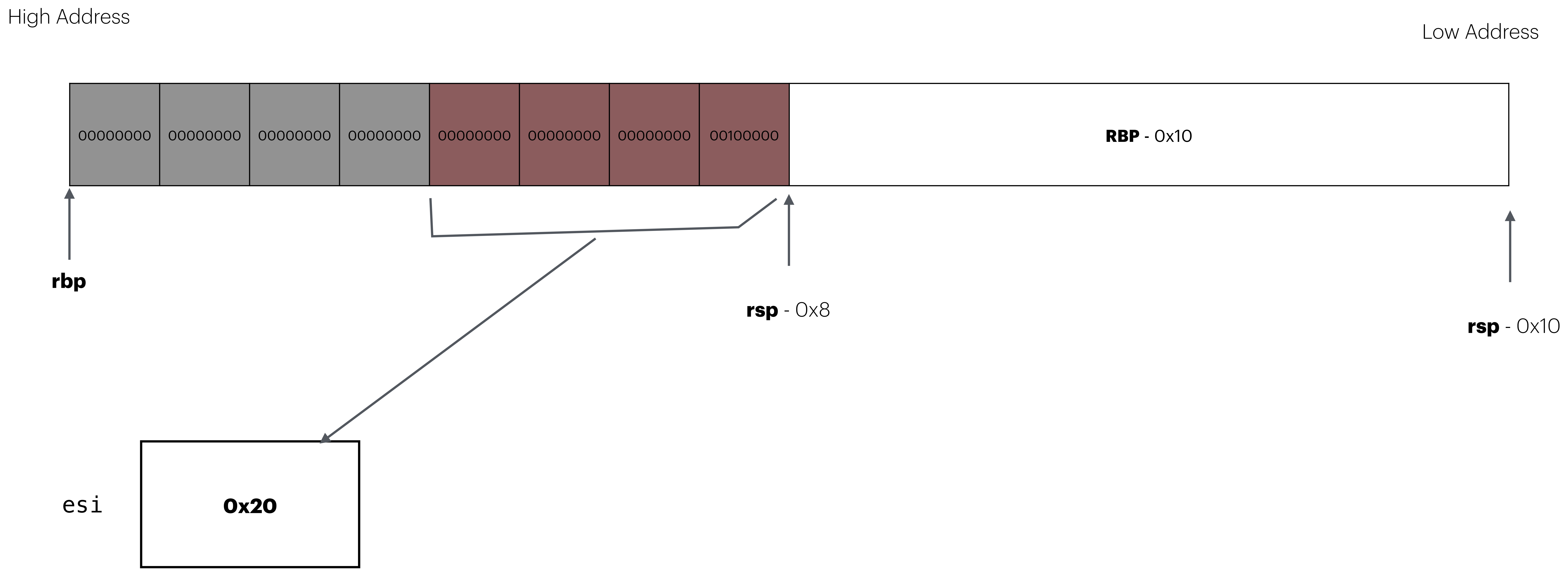
```
00000001000011f7 movl $0x20, -0x8(%rbp)
```



Load constant 0x20 (32) to stack location rbp - 0x8



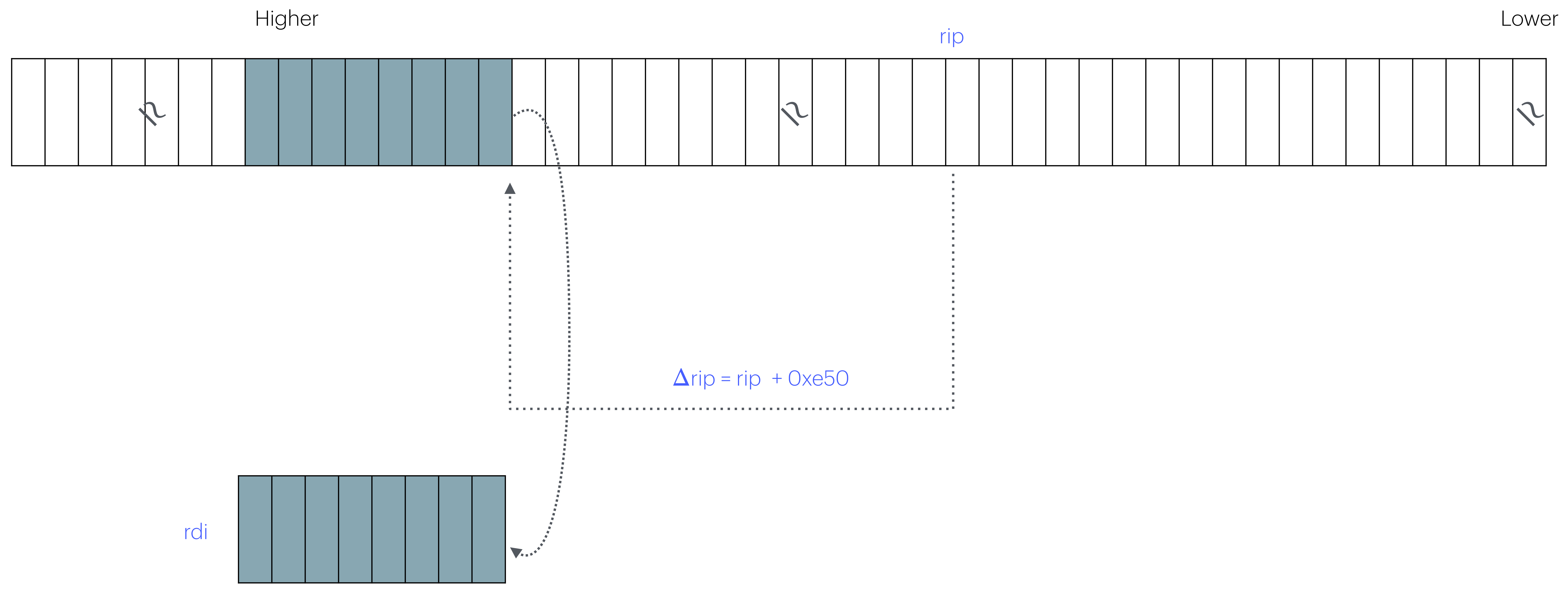
```
00000001000011fe movl -0x8(%rbp), %esi
```



**Transfer** 4 bytes of data from stack memory location (i.e. rsp - 0x8 ) into register esi

```
Note: esi is 32 bit register
```

```
0000000100001201 movq 0xe50(%rip), %rdi ## literal pool symbol address: __ZNSt3__14coutE
```

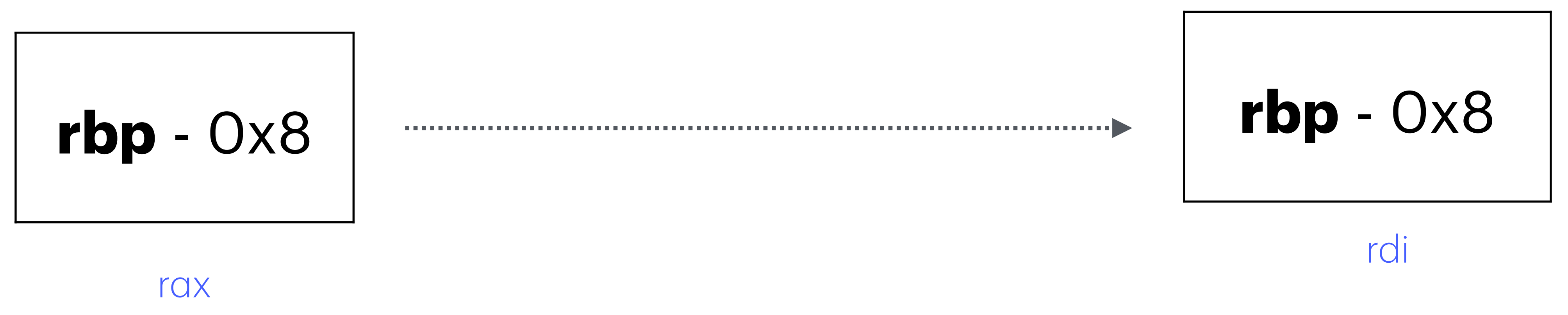


**Transfer** 8 bytes of data at memory location Δrip to register rdi

```
0000000100001208 callq  0x100001e58                    ## symbol stub for: __ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsEi
```

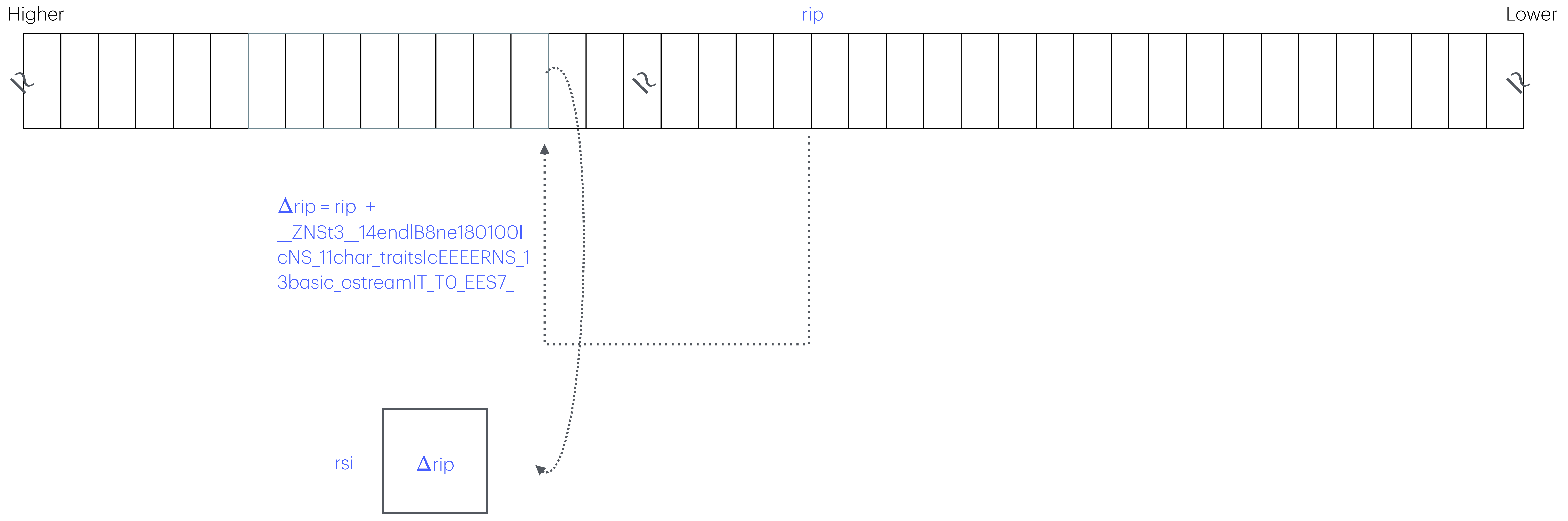
Calls std::cout function to print integer  value (i.e. 32 )

```
000000010000120d movq %rax, %rdi
```



Copies register rax into rdi ( sets the destination address, which is the stack address )

```
0000000100001210 leaq __ZNSt3__14endlB8ne180100IcNS_11char_traitsIcEEEERNS_13basic_ostreamIT_T0_EES7_(%rip),
%rsi ## std::__1::basic_ostream<char, std::__1::char_traits<char>>& std::__1::endl[abi:ne180100]<char,
std::__1::char_traits<char>>(std::__1::basic_ostream<char, std::__1::char_traits<char>>&)
```


Higher
rip
Lower
Δrip = rip + __ZNSt3__14endlB8ne180100IcNS_11char_traitsIcEEEERNS_13basic_ostreamIT_T0_EES7_
rsi
Δrip

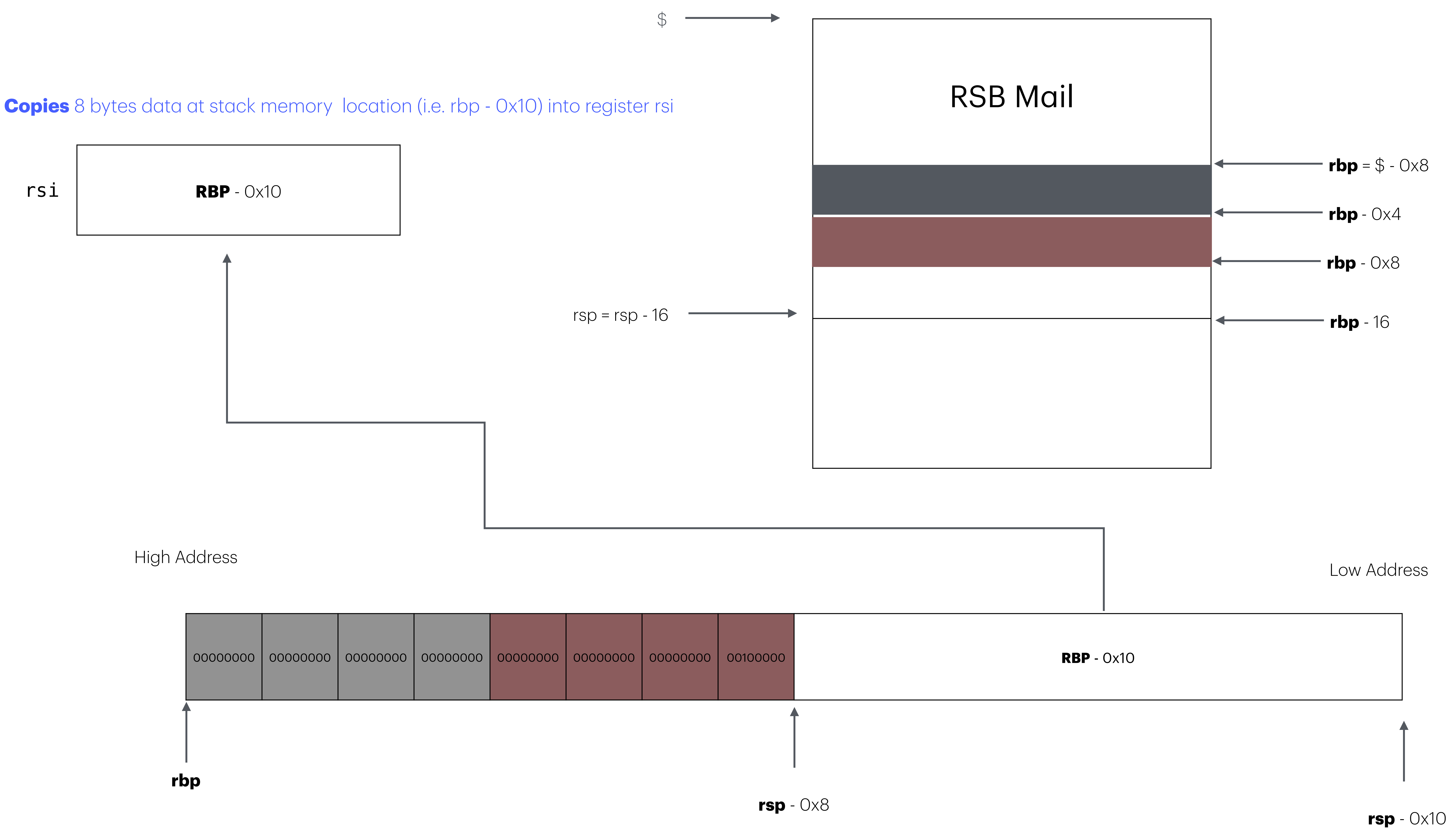
**Computes** memory location and loads (i.e. Δrip) into register rsi

```
0000000100001217 callq
__ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsB8ne180100EPFRS3_S4_E ##
std::__1::basic_ostream<char,
std::__1::char_traits<char>>::operator<<[abi:ne180100]
(std::__1::basic_ostream<char, std::__1::char_traits<char>>& (*)
(std::__1::basic_ostream<char, std::__1::char_traits<char>>&))
```

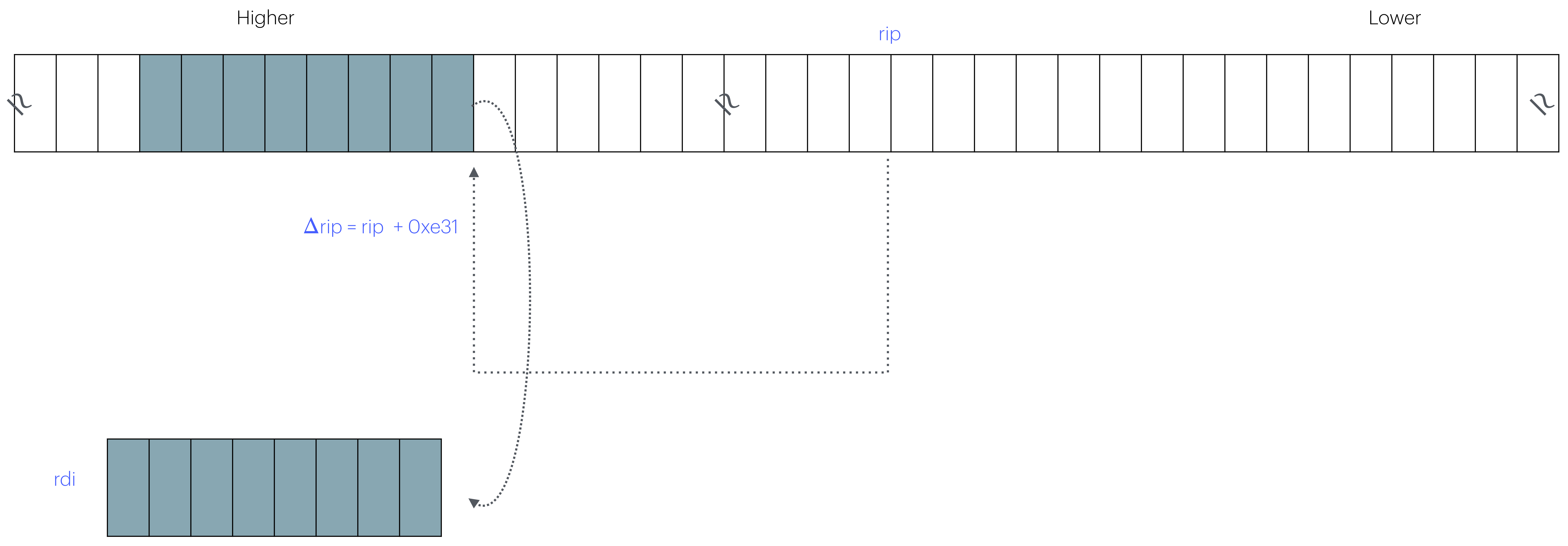
Calls std::cout function to print ptr  address

```
000000010000121c movq -0x10(%rbp), %rsi
```

**Copies** 8 bytes data at stack memory  location (i.e. rbp - 0x10) into register rsi



```
0000000100001220 movq 0xe31(%rip), %rdi   ## literal pool symbol address: __ZNSt3__14coutE
```

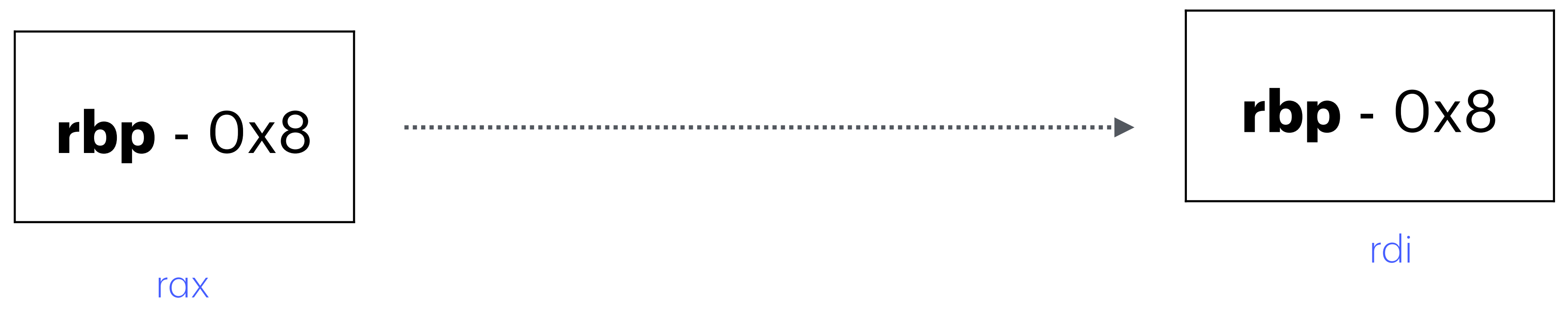


**Transfer** data ( 8 bytes) in memory location (i.e. Δrip) into register rdi

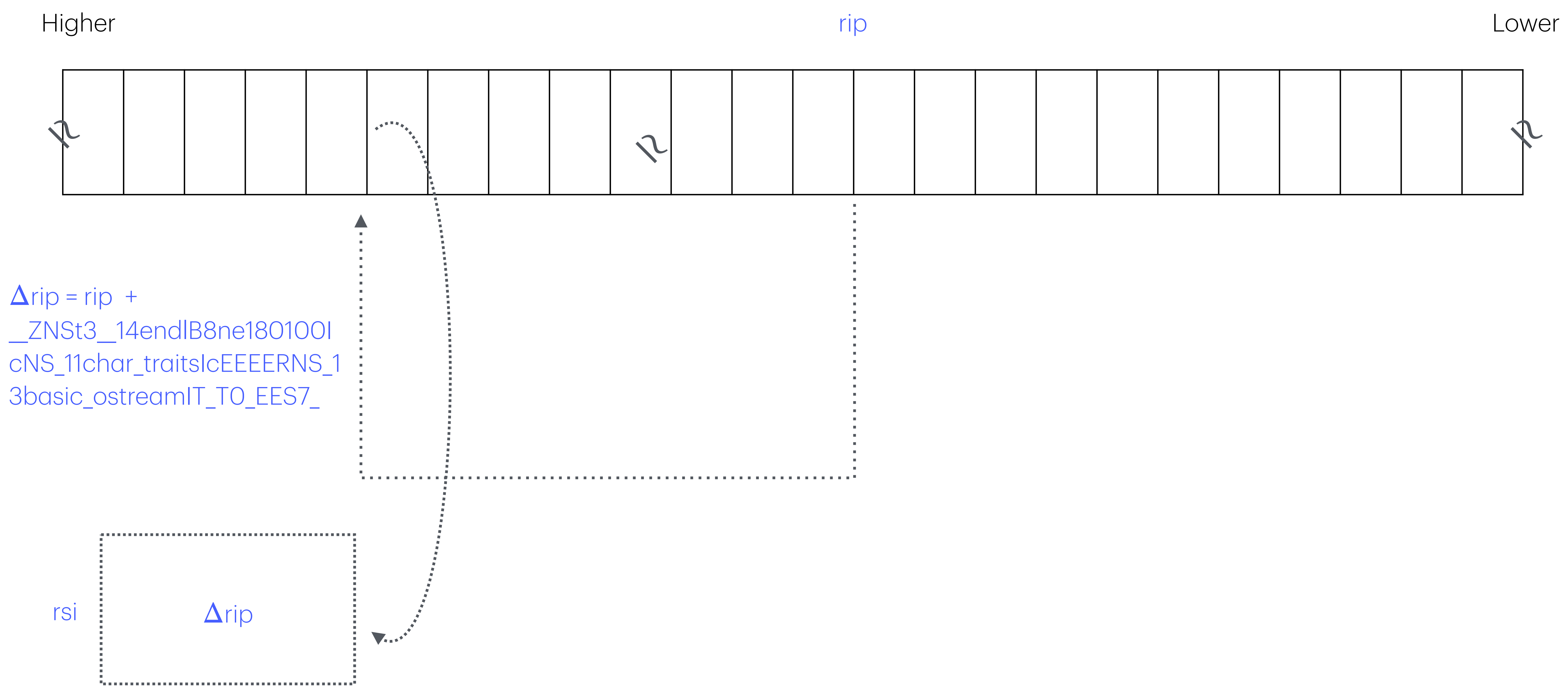
```
0000000100001227 callq  0x100001e52                          ## symbol stub for: __ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsEPKv
```

Caller

```
000000010000122c movq %rax, %rdi
```



```
000000010000122f  leaq  __ZNSt3__14endlB8ne180100IcNS_11char_traitsIcEEEERNS_13basic_ostreamIT_T0_EES7_(%rip), %rsi ##
std::__1::basic_ostream<char, std::__1::char_traits<char>>& std::__1::endl[abi:ne180100]<char,
std::__1::char_traits<char>>(std::__1::basic_ostream<char, std::__1::char_traits<char>>&)
```

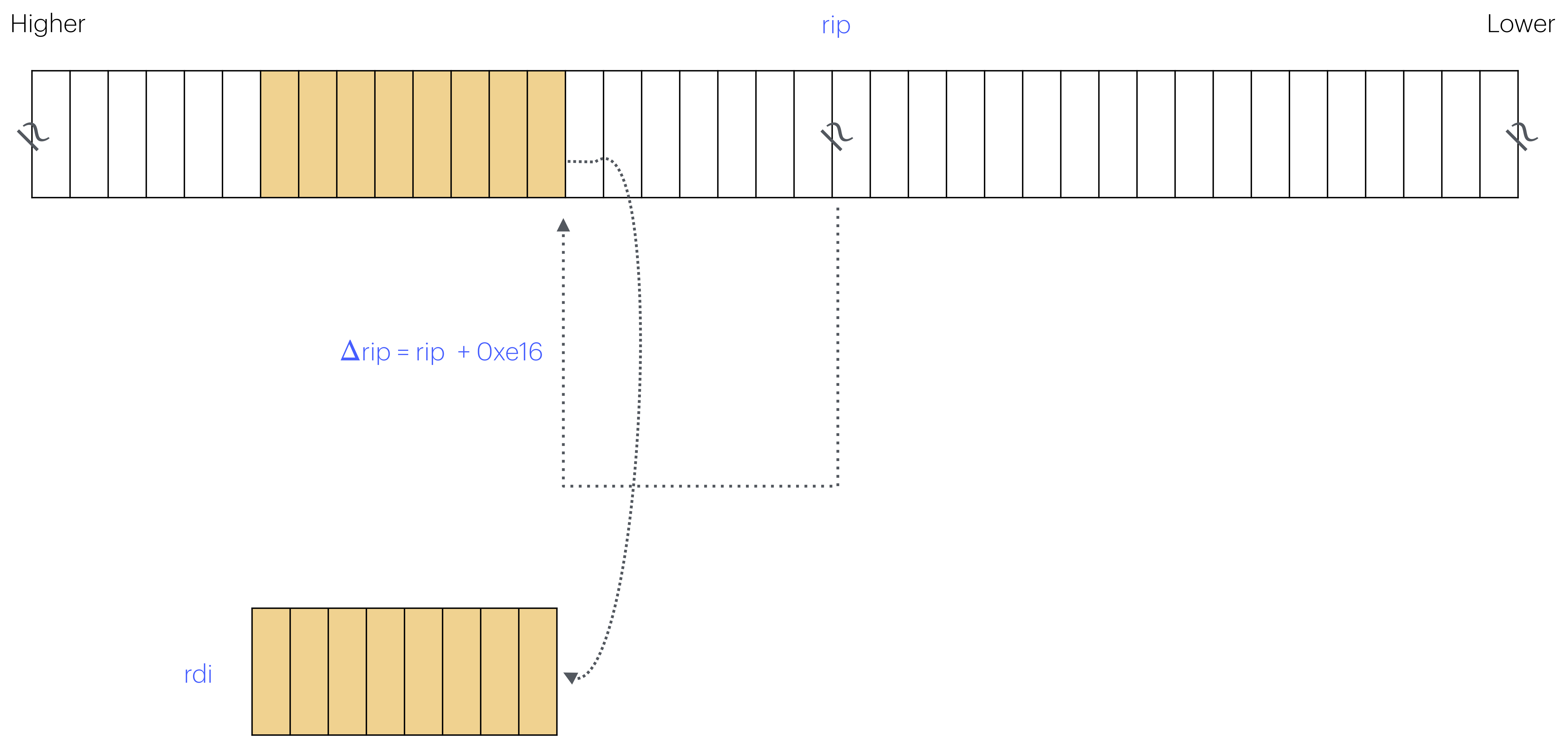

Higher
rip
Lower
Δrip = rip + __ZNSt3__14endlB8ne180100IcNS_11char_traitsIcEEEERNS_13basic_ostreamIT_T0_EES7_
rsi
Δrip

**Computes** memory address and loads (i.e. Δrip) into register rsi

```
0000000100001236  callq__ZNSt3__113basic_ostreamIcNS_11char_traitsIcEEElsB8ne180100EPFRS3_S4_E ##
std::__1::basic_ostream<char, std::__1::char_traits<char>>::operator<<[abi:ne180100](std::__1::basic_ostream<char,
std::__1::char_traits<char>>& (*)(std::__1::basic_ostream<char, std::__1::char_traits<char>>&))
```

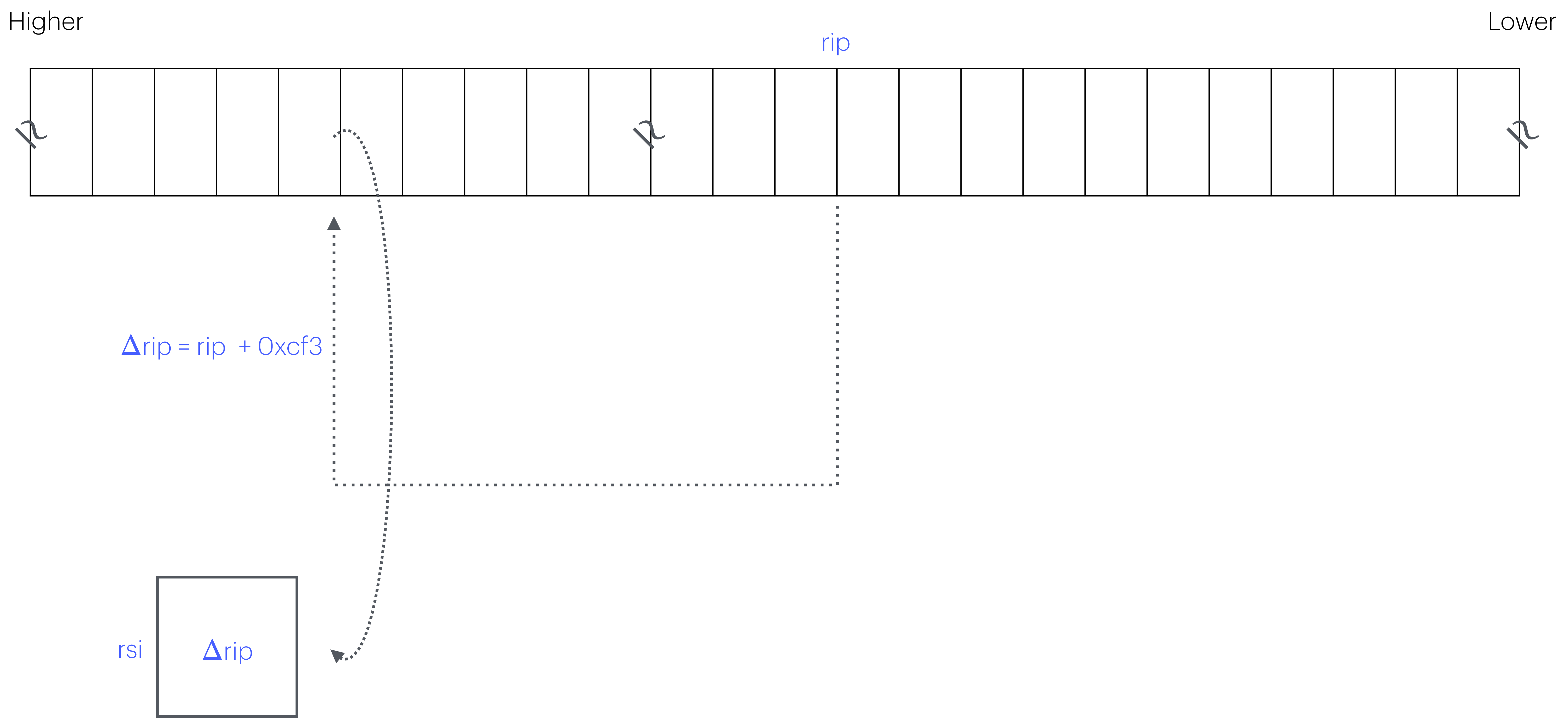
Caller prints ptr

```
000000010000123b  movq 0xe16(%rip), %rdi             ## literal pool symbol address: __ZNSt3__14coutE
```



**Transfer** 8 bytes from memory location Δrip into register rdi

```
0000000100001242   leaq 0xcf3(%rip), %rsi                ## literal pool for: "hello world."
```



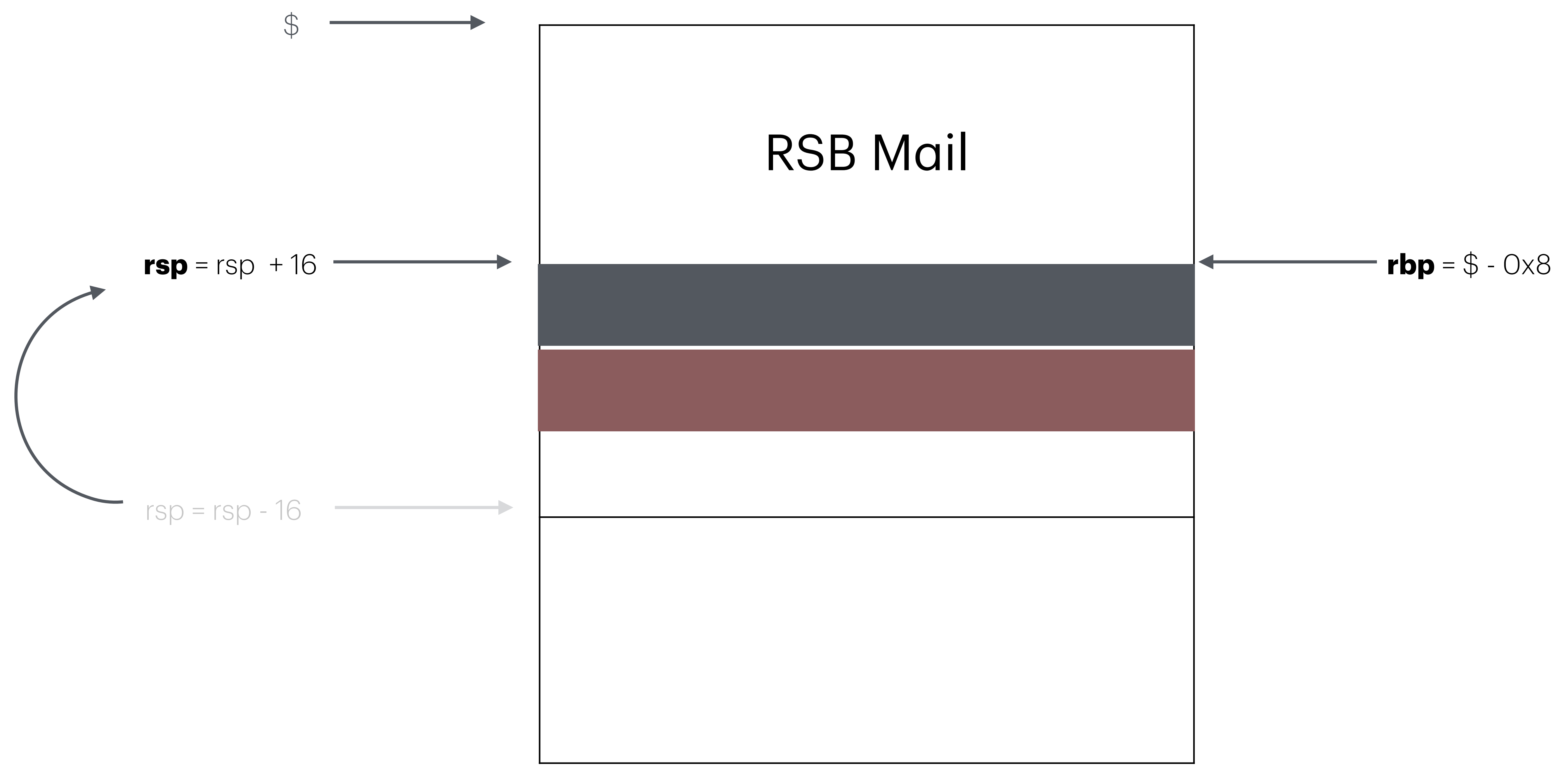
**Computes** memory location and loads (i.e. Δrip) into register rsi

```
0000000100001249 callq
__ZNSt3__1lsB8ne180100INS_11char_traitsIcEEEERNS_13basic_ostreamIcT_EES6_PKc ##
std::__1::basic_ostream<char, std::__1::char_traits<char>>&
std::__1::operator<<[abi:ne180100]<std::__1::char_traits<char>>(std::__1::basic_ostream<char,
std::__1::char_traits<char>>&, char const*)
```

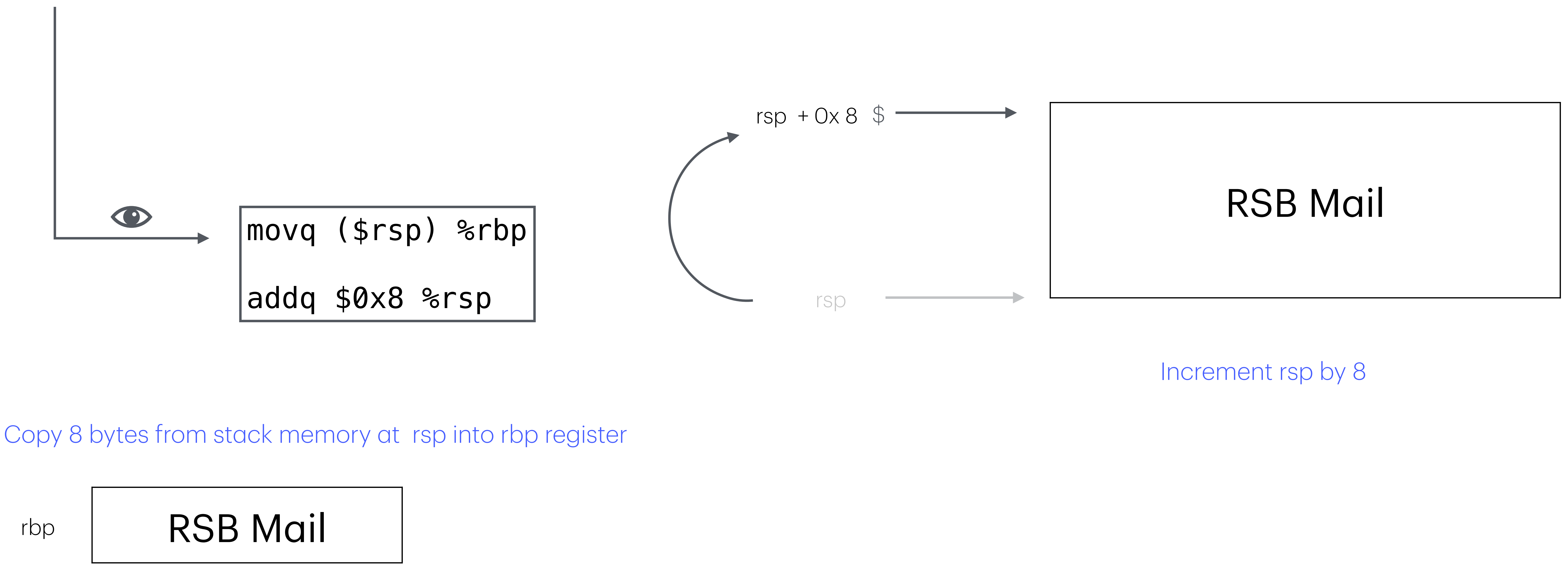
Call prints “hello world.”

```
000000010000124e xorl %eax, %eax
```

```
0000000100001250 addq $0x10, %rsp
```



```
0000000100001254 popq %rbp
```



```
movq ($rsp) %rbp

addq $0x8 %rsp
```

Increment rsp by 8

Copy 8 bytes from stack memory at rsp into rbp register