

## Re: Buffers in Assembly (NASM)

---

*Source:* <http://coding.derkeiler.com/Archive/Assembler/comp.lang.asm.x86/2008-07/msg00106.html>

---

- *From:* "Benjamin David Lunt" <[spamtrap@xxxxxxxxxx](mailto:spamtrap@xxxxxxxxxx)>
  - *Date:* Sat, 19 Jul 2008 00:09:10 -0700
- 

"bwaichu@xxxxxxxxxx" <[spamtrap@xxxxxxxxxx](mailto:spamtrap@xxxxxxxxxx)> wrote in message  
[news:9eb0b7cb-2b47-4e28-aad8-0cdc3a6b5a77@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:news:9eb0b7cb-2b47-4e28-aad8-0cdc3a6b5a77@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx)

I'm trying to better understand data structures in assembly. I know I can create a zero filled buffer in the bss section in NASM with this:

```
buffer: times 64 db 0
```

I am unfamiliar with NASM and I think it also depends on the object format you are using, but even if you put a zero as in above, doesn't the bss section still need to be zero'd? The above line will only occupy 64 bytes, it won't set it to zero. Am I correct?

And I know I can create a buffer on the stack like this:

```
sub esp, 64  
mov ebx, esp ;save the start point of the buffer
```

But how do I zero out the buffer on the stack? In C, I would just do something like:

```
char buffer[64] = {0};
```

Use your favorite C Compiler and compile this. Then either tell the compiler to output Assembly code, or disassemble it yourself and see what the compiler did.

What's the equivalent in assembly using NASM?

There are many ways. (I don't use NASM, but you should be able to convert it easy enough)

1.

Re: Buffers in Assembly (NASM)

```
mov ecx,((64+3)>>2)
xor eax,eax
@@: push eax
dec ecx
jnz short @b
```

```
2.
sub esp,((64+3) & ~3)
xor eax,eax
...
mov [esp+n],eax
mov [esp+n+4],eax
mov [esp+n+8],eax
...
```

```
3.
sub esp,((64+3) & ~3)
```

```
mov ecx,((64+3)>>2)
xor eax,eax
mov ebp,esp
@@: mov [ebp+n],eax
add ebp,4
dec ecx
jnz short @b
```

```
4.
sub esp,((64+3) & ~3)
mov ebx,esp
push byte 64
push byte 0
push ebx
call memset ; assuming your DS == SS
```

```
5.
your favorite code goes here
```

Ben

--

-----

Forever Young Software

<http://www.frontiernet.net/~fys/index.htm>

To reply by email, please remove the zzzzzz's

Batteries not included, some assembly required.

.