

Hello,

I think I found a BUG or may be I am wrong, and it's just something in the settings of MASM which I didn't use correctly. Below you will see a code snippet, where I try to load some pointer of procs which I have created in the same code seg ( there is only one because of 64 Bit and non segments only 1 flat). Regarding to the recommended software optimization guide of intel and AMD I aligned nearly all procedures on 16 Byte boundaries. As you can see in the picture (1) below the first offset calc (GREEN) works fine. This is due to adding a LABEL directive before the PROCEDURE prcSTR\_AVX512BlockCopy16 definition. ( you will see this view pages later. ) But the second one (RED) is wrong, as you can see on the address used for offset xxxxxx1032 which is not a natural 16 Byte boundary.

(Picture 1)

```
1358 mov r0, name_r0
1359 xor r9, r9
1360 mov rax, LSTR_AVX512BlockCopy16
1361 @@:
1362 shl r9, 5
1363 mov [rsp][r9], rcx
1364 mov [rsp][r9+08h], rdx
1365 mov [rsp][r9+10h], r8
1366 mov [rsp][r9+18h], rax
1367 shr r9, 5
1368 inc r9
1369 add rcx, rdx
1370 add r8, rdx
1371 cmp r9, QWORD PTR LOCAL_VAR1
1372 jnz @B
1373
1374 mov LOCAL_VAR7, rcx
1375 mov LOCAL_VAR8, r8
1376 xor rax, rax
1377 mov LOCAL_VAR6, rax
1378
1379 @@:
1380 mov rax, LOCAL_VAR6
1381 shl rax, 5
1382 xor rcx, rcx
1383 xor rdx, rdx
1384 lea r8, prcSTR_AVX512CopyBlock_T
1385 mov r8, prcSTR_AVX512CopyBlock_T
1386 ;;; mov r8, offset prcSTR_AVX512CopyBlock16_1
1387 mov r9, rsp
```

Assembly view details:

- Address 00007FFE50B82256: `mov r8, qword ptr [rbp+60h]`
- Address 00007FFE50B8225A: `xor r9, r9`
- Address 00007FFE50B8225D: `mov rax, offset prcSTR_AVX512BlockCopy16 (07FFE50B812E0h)` (Green circle)
- Address 00007FFE50B8228E: `xor rcx, rcx`
- Address 00007FFE50B822AE: `lea r8, [prcSTR_AVX512CopyBlock_T (07FFE50B81032h)]` (Red circle)
- Address 00007FFE50B822B5: `mov r8, offset prcSTR_AVX512CopyBlock_T (07FFE50B81032h)` (Red circle)

The following picture (2) shows you the definition of the correctly measured pointer (GREEN), which is those one where I've used the directive LABEL explicitly before the PROC definition. There is no bug on the align directive because as you can see the NOP which is included in the flow to start the PROC on 16 Byte boundary xxxxxx12E0 .

(Picture 2)

```
304 ;;;
305 ;;; Params: rcx = Pointer to Source
306 ;;;         rdx = Count bytes to copy (size)
307 ;;;         r8  = Pointer to dest
308 ;;; RetValue: count bytes copied
309 ;;;
310 ;;; Enrolling the loop
311 ;;;
312 align 10h
313 LSTR_AVX512BlockCopy16 LABEL NEAR
314 prcSTR_AVX512BlockCopy16 PROC
315 @Start:
316     PROLOG, XMMWORD
317     ;;; ALIGN_STACK XMMWORD
318     xor rax, rax
319     sub rsp, 1 * XMMWORD
320
321     vmovdqa xmm0, XMMWORD PTR[rcx]
322     vmovntdq XMMWORD PTR[r8], xmm0
323     mov rax, 1 * XMMWORD
324     cmp rdx, 1 * XMMWORD
```

Anzeigeoptionen

00007FFE50B812CE	4C 03 C0	add	r8,rax
00007FFE50B812D1	48 2B D0	sub	rdx,rax
00007FFE50B812D4	75 EA	jne	prcSTR_AVX512CopyBlock_T+20h (07FFE50B812C0h)
00007FFE50B812D6	48 83 C4 20	add	rsp,20h
00007FFE50B812DA	48 8B E5	mov	rsp,rbp
00007FFE50B812DD	5D	pop	rbp
00007FFE50B812DE	C3	ret	
00007FFE50B812DF	90	nop	
00007FFE50B812E0	55	push	rbp
00007FFE50B812E1	48 8B EC	mov	rbp,rsp
00007FFE50B812E4	48 83 E4 F0	and	rsp,0FFFFFFFFFFFFFF0h
00007FFE50B812E8	48 33 C0	xor	rax,rax
00007FFE50B812EB	48 83 EC 10	sub	rsp,10h
00007FFE50B812EF	C5 F9 6F 01	vmovdqa	xmm0,xmmword ptr [rcx]
00007FFE50B812F3	C4 C1 79 E7 00	vmovntdq	xmmword ptr [r8],xmm0
00007FFE50B812F8	48 C7 C0 10 00 00 00	mov	rax,10h
00007FFE50B812FF	48 83 FA 10	cmp	rdx,10h
00007FFE50B81303	0F 86 D6 00 00 00	jbe	prcSTR_AVX512BlockCopy16+0FFh (07FFE50B813DFh)
00007FFE50B81309	C5 F9 6F 49 10	vmovdqa	xmm1,xmmword ptr [rcx+10h]

And now in picture 3 I will show you the definition of the PROCEDURE which address is determined wrong. Again as you can see the align directive works fine (BLUE ; NOP etc..). And therefore the PROC is correctly aligned. But if you will remember the first picture (1) this address is not used for address calc ( see above pic. 1). The address which was determined is **07FFE50B81032h ( this is wrong )** it must be **07FFE50B812A0h ( this is the correct one )**

(Picture 3)

```

260 ;;; Copies x byte Blocks Threaded
261 ;;; Params: rcx = Pointer to Source
262 ;;;      rdx = Count bytes to copy (size)
263 ;;;      r8 = Pointer to Dest
264 ;;;      r9 = Pointer to BlockCopy procedure
265 ;;; RetValue: none
266 ;;;
267 -----
267 align 10h
268 procSTR_AVX512CopyBlock_T PROC
269 @Start:
270 PROLOG
271 sub rsp, 20h
272 mov HOME_RCX, rcx ;;; We save the original
273 mov rdx, [rcx+8]
274 mov r8, [rcx+10h]
275 mov r9, [rcx+18h]
276 mov rcx, [rcx] ;;; We load the source address
277 align 10h
278 @@:
279 sub rsp, 20h
280 call r9
281 add rsp, 20h
282 add rcx, rax
283 add r8, rax
284 sub rdx, rax
285 jnz @B
286 @Exit:
287 add rsp, 20h
288 EPILOG
289 ret
290 procSTR_AVX512CopyBlock_T ENDP

```

00007FFE50B81286	4C 03 CA	add	r9,rdx
00007FFE50B81289	49 8B C1	mov	rax,r9
00007FFE50B8128C	48 8B E5	mov	rsp,rbp
00007FFE50B8128F	48 83 C4 08	add	rsp,8
00007FFE50B81293	5D	pop	rbp
00007FFE50B81294	C3	ret	
00007FFE50B81295	66 66 66 0F 1F 84 00 00 00 00	nop	word ptr [rax+rax]
00007FFE50B812A0	55	push	rbp
00007FFE50B812A1	48 8B EC	mov	rbp,rsp
00007FFE50B812A4	48 83 E4 F0	and	rsp,0FFFFFFFFFFFFFFF0h
00007FFE50B812A8	48 83 EC 20	sub	rsp,20h
00007FFE50B812AC	48 89 4D 10	mov	qword ptr [rbp+10h],rcx
00007FFE50B812B0	48 8B 51 08	mov	rdx,qword ptr [rcx+8]
00007FFE50B812B4	4C 8B 41 10	mov	r8,qword ptr [rcx+10h]
00007FFE50B812B8	4C 8B 49 18	mov	r9,qword ptr [rcx+18h]
00007FFE50B812BC	48 8B 09	mov	rcx,qword ptr [rcx]
00007FFE50B812BF	90	nop	
00007FFE50B812C0	48 83 EC 20	sub	rsp,20h
00007FFE50B812C4	41 FF D1	call	r9
00007FFE50B812C7	48 83 C4 20	add	rsp,20h
00007FFE50B812CB	48 03 C8	add	rcx,rax
00007FFE50B812CE	4C 03 C0	add	r8,rax
00007FFE50B812D1	48 2B D0	sub	rdx,rax
00007FFE50B812D4	75 EA	jne	procSTR_AVX512CopyBlock_T+20h (07FFE50B812C0h)
00007FFE50B812D6	48 83 C4 20	add	rsp,20h
00007FFE50B812DA	48 8B E5	mov	rsp,rbp
00007FFE50B812DD	5D	pop	rbp
00007FFE50B812DE	C3	ret	
00007FFE50B812DF	90	nop	
00007FFE50B812E0	55	push	rbp
00007FFE50B812E1	48 8B EC	mov	rbp,rsp
00007FFE50B812E4	48 83 E4 F0	and	rsp,0FFFFFFFFFFFFFFF0h

Instead of using the correct address, I find out that the compiler creates something like a table where all the procedure are available (Picture 4). This looks like a dynamic created table of pointer's to all procs of the module. I didn't create this.

I hope there is only something missing or wrong defined in the settings of Visual Studio or MASM.

(Picture 4)

```
00007FFE50B81005 E9 A6 0D 00 00 jmp prcSTR_AVX512ProcessCopyBlock16 (07FFE50B81DB0h)
00007FFE50B8100A E9 11 14 00 00 jmp prcSTR_AVX512CopyBlock16_T (07FFE50B82420h)
00007FFE50B8100F E9 DC 03 00 00 jmp prcSTR_AVX512BlockCopy32 (07FFE50B813F0h)
00007FFE50B81014 E9 F7 0D 00 00 jmp prcSTR_AVX512ExecCopyBlock64_T (07FFE50B81E10h)
00007FFE50B81019 E9 62 0C 00 00 jmp prcSTR_AVX512CopyString (07FFE50B81C80h)
00007FFE50B8101E E9 AD 0F 00 00 jmp prcSTR_AVX512ExecCopyBlock32_T (07FFE50B81FD0h)
00007FFE50B81023 E9 D0 00 00 00 jmp prcSTR_AVX512GetLengthAligned16 (07FFE50B810F8h)
00007FFE50B81028 E9 23 0D 00 00 jmp prcSTR_AVX512ProcessCopyBlock32 (07FFE50B81D50h)
00007FFE50B8102D E9 46 14 00 00 jmp prcSTR_AVX512StringEqual (07FFE50B82478h)
00007FFE50B81032 E9 69 02 00 00 jmp prcSTR_AVX512CopyBlock_T (07FFE50B812A0h)
00007FFE50B81037 E9 04 13 00 00 jmp prcSTR_AVX512CopyBlock32_T (07FFE50B82360h)
00007FFE50B8103C E9 8F 00 00 00 jmp prcLib_AVX512Initialize (07FFE50B810D0h)
00007FFE50B81041 E9 4A 11 00 00 jmp prcSTR_AVX512ExecCopyBlock16_T (07FFE50B82190h)
00007FFE50B81046 E9 2D 01 00 00 jmp prcSTR_AVX512GetLength16 (07FFE50B81178h)
00007FFE50B8104B E9 10 13 00 00 jmp prcSTR_AVX512CopyBlock64_T (07FFE50B82360h)
00007FFE50B81050 E9 9B 0C 00 00 jmp prcSTR_AVX512ProcessCopyBlock64 (07FFE50B81CF0h)
00007FFE50B81055 E9 86 02 00 00 jmp prcSTR_AVX512BlockCopy16 (07FFE50B812E0h)
00007FFE50B8105A E9 D9 00 00 00 jmp prcSTR_AVX512GetLengthAligned32 (07FFE50B81138h)
00007FFE50B8105F E9 A4 01 00 00 jmp prcSTR_AVX512GetLength32 (07FFE50B81208h)
00007FFE50B81064 E9 37 06 00 00 jmp prcSTR_AVX512BlockCopy64 (07FFE50B816A0h)
```