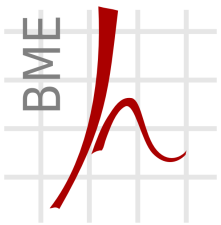
Ciklus –X87 – Objektum

Kódvisszafejtés.



Híradástechnikai Tanszék

Izsó Tamás

2012. november 8.

Section 1

Ciklus optimalizálás

Ciklus kiértékelése fordítási idő alatt

```
int main() {
    int i=0, s=0, k=5;
    for( i=0; i<k; i++) s = s+i;
    printf("%d\n", s );
    return 0;
}
```

```
_main:
   00000000 6A 0A              push 0Ah
   00000002 68 00 00 00 00     push offset ??_C@_03PMGGPEJJ@?$CFd?6?$AA@
   00000007 E8 00 00 00 00     call _printf
   0000000C 83 C4 08           add  esp,8
   0000000F 33 C0              xor  eax,eax
   00000011 C3                 ret
```

Section 2

x87 processzor

x87 processzor regiszterei

- regiszterek
  - 8 darab regiszter
  - stack szervezés
  - 32, 64 és 80 bites float számokat tartalmazhat
- önálló flag regiszter
- a stack tartalmát függvényhívások között megőrzi

x87 processzor adatmozgató utasítások

- FLD – Load Floating Point
- FST – Store Floating Point
- FSTP – Store Floating Point and Pop
- FXCH – Exchange Register Contents
- FILD – Load Integer

x87 processzor – fontosabb konstansok

- FLDZ Load +0.0
- FLD1 Load +1.0
- FLDPI Load π
- FLDL2T Load $log_2(10)$
- FLDL2E Load $log_2(e)$
- FLDLG2 Load $log_{10}(2)$
- FLDLN2 Load $log_e(2)$

x87 alaputasítások

- FADD/FADDP Add floating point
- FIADD Add integer to floating point
- FSUB/FSUBP Subtract floating point
- FISUB Subtract integer from floating point
- FSUBR/FSUBRP Reverse subtract floating point
- FISUBR Reverse subtract floating point from integer
- FMUL/FMULP Multiply floating point
- FIMUL Multiply integer by floating point
- FDIV/FDIVP Divide floating point
- FIDIV Divide floating point by integer
- FDIVR/FDIVRP Reverse divide
- FIDIVR Reverse divide integer by floating point
- FABS Absolute value
- FCHS Change sign
- FSQRT Square root
- FRNDINT Round to integral value

x87 reverse utasítás értelmezése

FSUB:

$$ST(0) \leftarrow ST(0) - ST(i)$$
$$ST(i) \leftarrow ST(i) - ST(0)$$

FSUBR:

$$ST(0) \leftarrow ST(i) - ST(0)$$
$$ST(i) \leftarrow ST(0) - ST(i)$$

Másodfokú egyenletmegoldó program

```
int masodfoku(double a, double b, double c,
              double* res1, double *res2)
{
    double disc;
    disc = b*b - 4*a*c;
    if( disc < 0 ) return 0;
    *res1 = (-b+sqrt(disc))/(2*a);
    *res2 = (-b-sqrt(disc))/(2*a);
    return 1;
}
```

Másodfokú egyenletmegoldó gépi kódja

```
_masodfoku:
  0000  fld     qword ptr [esp+0Ch]   st: b
  0004  fld     st(0)                 st: b, b
  0006  fmul    st,st(1)              st: b*b, b
  0008  fld     qword ptr [esp+4]     st: a, b*b, b
  000C  fld     qword ptr [esp+14h]   st: c, a, b*b, b
  0010  fmul    st,st(1)              st: c*a, a, b*b, b
  0012  fmul    qword ptr [__real_4]  st: 4*c*a, a, b*b, b
  0018  fsubp   st(2),st              st: a, b*b-4*a*c, b
  001A  fldz                          st: 0, a, b*b-4*a*c, b
  001C  fcomp   st(2)                 st: a, disc, b
  001E  fnstsw  ax                    ax <- status word
  0020  test    ah,41h
  0023  jne     0000002E              0 <= disc
  0025  fstp    st(1)                 st: disc, b
  0027  xor     eax,eax
  0029  fstp    st(1)                 st: b
  002B  fstp    st(0)                 st:
  002D  ret
  002E  fxch    st(1)                 st: disc, a, b
  0030  mov     eax,dword ptr [esp+1Ch]
  0034  fsqrt                         st: sqrt(disc), a, b
  0036  mov     ecx,dword ptr [esp+20h]
  003A  fxch    st(1)                 st: a, sqrt(disc), b
  003C  fadd    st(0),st              st: 2*a, sqrt(disc), b
  003E  fld     st(1)                 st: sqrt(disc), 2*a, sqrt(disc), b
  0040  fsub    st,st(3)              st: sqrt(disc)-b, 2*a, sqrt(disc), b
  0042  fdiv    st,st(1)              st: (sqrt(disc)-b)/ (2*a), 2*a, sqrt(disc), b
  0044  fstp    qword ptr [eax]       st: 2*a, sqrt(disc), b
  0046  fxch    st(2)                 st: b, sqrt(disc), 2*a
  0048  fchs                          st: -b, sqrt(disc), 2*a
  004A  mov     eax,1
  004F  fsubrp  st(1),st              st: -b-sqrt(disc), 2*a /reverse !!! rp
  0051  fdivrp  st(1),st              st: -b-sqrt(disc)/ 2*a
  0053  fstp    qword ptr [ecx]
  0055  ret
```

Másodfokú egyenletmegoldó gépi kódja – SSE3

```
_masodfoku:
   00000000: F2 0F 10 5C 24 04   movsd   xmm3,mmword ptr [esp+4]
   00000006: 66 0F EF C9         pxor    xmm1,xmm1
   0000000A: F2 0F 10 05000000   movsd   xmm0,mmword ptr [_2il0floatpacket.5]
   00000012: F2 0F 59 C3         mulsd   xmm0,xmm3
   00000016: F2 0F 10 64 24 0C   movsd   xmm4,mmword ptr [esp+0Ch]
   0000001C: 0F 28 D4            movaps  xmm2,xmm4
   0000001F: F2 0F 59 44 24 14   mulsd   xmm0,mmword ptr [esp+14h]
   00000025: F2 0F 59 D4         mulsd   xmm2,xmm4
   00000029: F2 0F 5C D0         subsd   xmm2,xmm0
   0000002D: 66 0F 2F CA         comisd  xmm1,xmm2
   00000031: 76 03               jbe     00000036
   00000033: 33 C0               xor     eax,eax
   00000035: C3                  ret
   00000036: F2 0F 51 D2         sqrtsd  xmm2,xmm2
   0000003A: F2 0F 58 DB         addsd   xmm3,xmm3
   0000003E: 0F 28 C2            movaps  xmm0,xmm2
   00000041: 66 0F EF C9         pxor    xmm1,xmm1
   00000045: 8B 44 24 1C         mov     eax,dword ptr [esp+1Ch]
   00000049: F2 0F 5C C4         subsd   xmm0,xmm4
   0000004D: F2 0F 58 E2         addsd   xmm4,xmm2
   00000051: F2 0F 5E C3         divsd   xmm0,xmm3
   00000055: 8B 54 24 20         mov     edx,dword ptr [esp+20h]
   00000059: F2 0F 5C CC         subsd   xmm1,xmm4
   0000005D: F2 0F 5E CB         divsd   xmm1,xmm3
   00000061: F2 0F 11 00         movsd   mmword ptr [eax],xmm0
   00000065: B8 01 00 00 00      mov     eax,1
   0000006A: F2 0F 11 0A         movsd   mmword ptr [edx],xmm1
   0000006E: C3                  ret
```

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

????? sum(?????) {

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

```
????? sum( ????? ) {
    dword t1 = args1;
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

```
????? sum( dword args1 ) {
        dword t1 = args1;
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

```
????? sum( dword args1 ) {
        dword t1 = args1;
        dword t2 = *(dword*)t1;
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

```
????? sum( dword* args1 ) {
        dword t1 = args1;
        dword t2 = *(dword*)t1;
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

```
????? sum( dword* args1 ) {
        dword t1 = args1;
        dword t2 = *(dword*)t1;
        t2 = t2 +10;
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

```
????? sum( dword* args1 ) {
        dword t1 = args1;
        dword t2 = *(dword*)t1;
        t2 = t2 +10;
        return t2;
}
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Típus visszaállítás

```
_sum:
    mov   edx,dword ptr [esp+4]
    mov   eax,dword ptr [edx]
    add   eax,0Ah
    ret
    lea   esi,[esi+00000000h]
```

```
dword sum( dword* x ) {
        dword t1 = args1;
        dword t2 = *(dword*)t1;
        t2 = t2 +10;
        return t2;
}
```

Továbbra sem ismert a visszatérési érték típusa, de azt tudjuk, hogy 32 bites numerikus típus.

Globális adatok inicializálása

```
char c;
int  i;
short int h;
int p1;
int p2;
int p3;
void init()
{
    c = 'A';
    h = 10;
    i = 1024;
    p1 = 0x401000;
    p2 = 0x401003;
    p3 = 0x401005;
}
```

Globális adatok inicializálása

```
00401000                                        public start
00401000                                start   proc near
00401000 B8 0A 00 00 00                         mov   eax, 0Ah
00401005
00401005                                loc_401005:
00401005 C6 05 00 20 40 00 41                   mov   byte_402000, 41h
0040100C 66 A3 02 20 40 00                      mov   word_402002, ax
00401012 C7 05 04 20 40 00 00 04 00 00          mov   dword_402004, 400h
0040101C C7 05 08 20 40 00 00 10 40 00          mov   dword_402008, offset start
00401026 C7 05 0C 20 40 00 03 10 40 00          mov   dword_40200C, 401003h
00401030 C7 05 10 20 40 00 05 10 40 00          mov   dword_402010, offset loc_401005
0040103A C3                                     retn
0040103A                                start   endp
```

```
00402000 ??                             byte_402000     db ?
00402001 ??                                             align 2
00402002 ?? ??                          word_402002     dw ?
00402004 ?? ?? ?? ??                    dword_402004    dd ?
00402008 ?? ?? ?? ??                    dword_402008    dd ?
0040200C ?? ?? ?? ??                    dword_40200C    dd ?
00402010 ?? ?? ?? ??                    dword_402010    dd ?
00402010                                _data           ends
```

Globális adatok inicializálása

- karakter típusú adatra: **mov** byte_402000, 41h
- short int típusú adatra: **mov** word_402002, **ax**
- int típusú adatra: **mov** dword_402004, 400h
- Ha a kódszegmensben az adott címen utasítás van, akkor az IDA *beugrik*. Pl.
  - **mov** dword_402008, **offset** start
  - **mov** dword_402010, **offset** loc_401005

Strukturák visszaadása visszatérési értékként

```
#include <stdio.h>
typedef struct A_struct  {
    int x;
    int y;
    int z;
} A;

A init_A(int value) {
    A v;
    v.x = 10;
    v.y = 20;
    v.z = 30;

    return v;
}

int main() {
    A r;
     r = init_A(5);
    printf("(%d,%d,%d)",r.x,r.y,r.z);
    return 0;
}
```

Strukturák visszaadása visszatérési értékként

```
#include <stdio.h>
typedef struct A_struct  {
    int x;
    int y;
    int z;
} A;

A init_A(int value) {
    A v;
    v.x = 10;
    v.y = 20;
    v.z = 30;

    return v;
}

int main()
    A r;
     r = init_A(5);
    printf("(%d,%d,%d)",r.x,r.y,r.z);
    return 0;
}
```

Lokális struktúra inicializálása.

Strukturák visszaadása visszatérési értékként

```
#include <stdio.h>
typedef struct A_struct  {
    int x;
    int y;
    int z;
} A;

A init_A(int value) {
    A v;
    v.x = 10;
    v.y = 20;
    v.z = 30;

    return v;
}

int main() {
    A r;
     r = init_A(5);
    printf("(%d,%d,%d)",r.x,r.y,r.z);
    return 0;
}
```

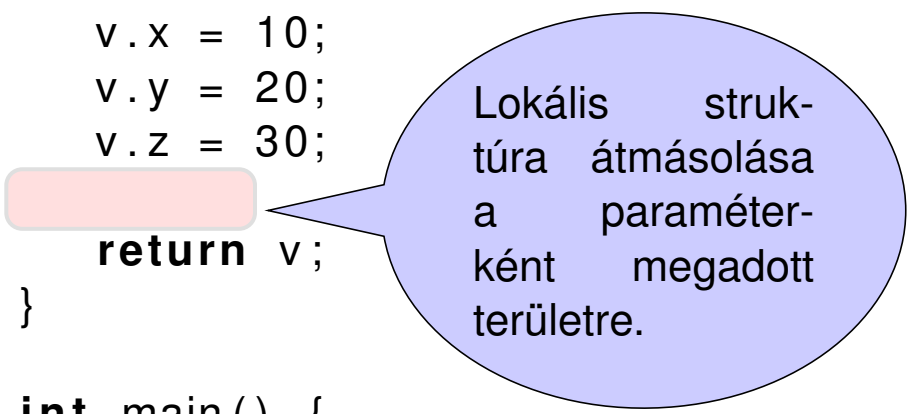
Visszatérési érték 0. paraméterként kerül átadásra.

Strukturák visszaadása visszatérési értékként

```
#include <stdio.h>
typedef struct A_struct  {
    int x;
    int y;
    int z;
} A;

A init_A(int value) {
    A v;
    v.x = 10;
    v.y = 20;
    v.z = 30;
    return v;
}

int main() {
    A r;
     r = init_A(5);
    printf("(%d,%d,%d)",r.x,r.y,r.z);
    return 0;
}
```



Strukturák visszaadása visszatérési értékként

```
#include <stdio.h>
typedef struct A_struct  {
    int x;
    int y;
    int z;
} A;

A init_A(int value) {
    A v;
    v.x = 10;
    v.y = 20;
    v.z = 30;

    return v;
}

int main() {
    A r;
    r = init_A(5);
    printf("(%d,%d,%d)",r.
    return 0;
}
```

Visszaadott érték átmásolása r-be.

Strukturák visszaadása visszatérési értékként

```
0000000C _init_A    proc near
0000000C            push    ebp
0000000D            mov     ebp, esp
0000000F            sub     esp, 0Ch
00000012            mov     eax, [ebp+0Ch]
00000015            mov     [ebp - 0Ch], eax
00000018            mov     [ebp - 8], 14h
0000001F            mov     [ebp- 4], 1Eh
00000026            mov     ecx, [ebp+8]
00000029            mov     edx, [ebp- 0Ch]
0000002C            mov     [ecx], edx
0000002E            mov     eax, [ebp - 8]
00000031            mov     [ecx+4], eax
00000034            mov     edx, [ebp - 4]
00000037            mov     [ecx+8], edx
0000003A            mov     eax, [ebp + 8]
0000003D            mov     esp, ebp
0000003F            pop     ebp
00000040            retn
00000040 _init_A        endp
```

Strukturák visszaadása visszatérési értékként

```
0000000C _init_A    proc
0000000C            push
0000000D            mov
0000000F            sub
00000012            mov     eax, [ebp+0Ch]
00000015            mov     [ebp - 0Ch], eax
00000018            mov     [ebp - 8], 14h
0000001F            mov     [ebp- 4], 1Eh
00000026            mov     ecx, [ebp+8]
00000029            mov     edx, [ebp- 0Ch]
0000002C            mov     [ecx], edx
0000002E            mov     eax, [ebp - 8]
00000031            mov     [ecx+4], eax
00000034            mov     edx, [ebp - 4]
00000037            mov     [ecx+8], edx
0000003A            mov     eax, [ebp + 8]
0000003D            mov     esp, ebp
0000003F            pop     ebp
00000040            retn
00000040 _init_A    endp
```

r változó iniciali-
zálása.

Strukturák visszaadása visszatérési értékként

```
0000000C _init_A    proc near
0000000C            push
0000000D            m
0000000F            s
00000012            m
00000015            mov
00000018            mov     [ebp      ], 14h
0000001F            mov     [ebp-     4], 1Eh
00000026            mov     ecx, [ebp+8]
00000029            mov     edx, [ebp- 0Ch]
0000002C            mov     [ecx], edx
0000002E            mov     eax, [ebp - 8]
00000031            mov     [ecx+4], eax
00000034            mov     edx, [ebp - 4]
00000037            mov     [ecx+8], edx
0000003A            mov     eax, [ebp + 8]
0000003D            mov     esp, ebp
0000003F            pop     ebp
00000040            retn
00000040 _init_A    endp
```

r változó tartalmának az átmásolása a 0. paraméterben megadott helyre.

Strukturák visszaadása visszatérési értékként

```
_main        proc near
0000004C             push    ebp
0000004D             mov     ebp, esp
0000004F             sub     esp, 3
00000052             and     esp, 0FFFFFFF8h
00000055             add     esp, 4
00000058             sub     esp, 18h
0000005B             lea     eax, dword ptr [ebp - 10h]
0000005E             mov     dword ptr [esp], eax
00000061             mov     dword ptr [esp+4], 5
00000069             call    _init_A
0000006E             add     esp, 8
00000071             add     esp, 0FFFFFFF0h
00000074             mov     dword ptr [esp], offset aDDD ; "(%d,%d,%d)"
0000007B             mov     eax, dword ptr [ebp - 10h]
0000007E             mov     dword ptr [esp+4], eax
00000082             mov     eax, dword ptr [ebp - 0Ch]
00000085             mov     dword ptr [esp+8], eax
00000089             mov     eax, dword ptr [ebp - 8h]
0000008C             mov     dword ptr [esp+0Ch], eax
00000090             call    _printf
00000095             add     esp, 10h
00000098             mov     dword ptr [ebp - 4h], eax
0000009B             mov     eax, 0
000000A0             leave
000000A1             retn
000000A1 _main       endp
```

Strukturák visszaadása visszatérési értékként

```
_main        proc near
0000004C             push    ebp
0000004D             mov     ebp, 
0000004F             sub     esp, 
00000052             and     esp, 0
00000055             add     esp, 4
00000058             sub     esp, 18h
0000005B             lea     eax, dword ptr [ebp - 10h]
0000005E             mov     dword ptr [esp], eax
00000061             mov     dword ptr [esp+4], 5
00000069             call    _init_A
0000006E             add     esp, 8
00000071             add     esp, 0FFFFFFF0h
00000074             mov     dword ptr [esp], offset aDDD ; "(%d,%d,%d)"
0000007B             mov     eax, dword ptr [ebp - 10h]
0000007E             mov     dword ptr [esp+4], eax
00000082             mov     eax, dword ptr [ebp - 0Ch]
00000085             mov     dword ptr [esp+8], eax
00000089             mov     eax, dword ptr [ebp - 8h]
0000008C             mov     dword ptr [esp+0Ch], eax
00000090             call    _printf
00000095             add     esp, 10h
00000098             mov     dword ptr [ebp - 4h], eax
0000009B             mov     eax, 0
000000A0             leave
000000A1             retn
000000A1 _main       endp
```

r struktúra címének a stackre helyezése.

Strukturák visszaadása visszatérési értékként

```
_main        proc near
0000004C             push    ebp
0000004D             mov     ebp, esp
0000004F             sub     esp, 3
00000052             and     esp, 0FFFFFFF8h
00000055             add     esp, 4
00000058             sub     esp, 18h
0000005B             lea     eax, dwor                10h]
0000005E             mov     dword
00000061             mov     dword
00000069             call    _init_
0000006E             add     esp, 8
00000071             add     esp, 0FFFFFF
00000074             mov     dword ptr [esp]  offset aDDD ; "(%d,%d,%d)"
0000007B             mov     eax, dword ptr [ebp - 10h]
0000007E             mov     dword ptr [esp+4], eax
00000082             mov     eax, dword ptr [ebp - 0Ch]
00000085             mov     dword ptr [esp+8], eax
00000089             mov     eax, dword ptr [ebp - 8h]
0000008C             mov     dword ptr [esp+0Ch], eax
00000090             call    _printf
00000095             add     esp, 10h
00000098             mov     dword ptr [ebp - 4h], eax
0000009B             mov     eax, 0
000000A0             leave
000000A1             retn
000000A1 _main       endp
```

r változó értéké-nek a stackre he-lyezése a kiíra-táshoz.

Strukturák visszaadása visszatérési értékként (optimalizált 1.)

```
00000000 _main          proc near
00000000 var_80         = dword ptr -80h
00000000                push    ebp
00000001                mov     ebp, esp
00000003                and     esp, 0FFFFFF80h
00000006                sub     esp, 80h
0000000C                push    3
0000000E                call    ___intel_new_proc_init
00000013                stmxcsr [esp+84h+var_80]
00000018                or      [esp+84h+var_80], 8000h
00000020                ldmxcsr [esp+84h+var_80]
00000025                push    1Eh[1]
00000027                push    14h
00000029                push    5
0000002B                push    offset ??_C@_0L@A@?$CJ?$AA@ ; "(%d,%d,%d)"
00000030                call    _printf
00000035                xor     eax, eax
00000037                mov     esp, ebp
00000039                pop     ebp
0000003A                retn
0000003A _main          endp
```

[1] Inline hívás + optimalizálás.

Strukturák visszaadása visszatérési értékként (optimalizált 1.)

```
00000000 _main          proc near
00000000 var_80         =
00000000
00000001
00000003                                        FF80h
00000006                s
0000000C                push
0000000E                call    __intel_new_proc_init
00000013                stmxcsr [esp+84h+var_80]
00000018                or      [esp+84h+var_80], 8000h
00000020                ldmxcsr [esp+84h+var_80]
00000025                push    1Eh[1]
00000027                push    14h
00000029                push    5
0000002B                push    offset ??_C@_0L@A@?$CJ?$AA@ ; "(%d,%d,%d)"
00000030                call    _printf
00000035                xor     eax, eax
00000037                mov     esp, ebp
00000039                pop     ebp
0000003A                retn
0000003A _main          endp
```

Flush to Zero – SSE lebegő-pontos számítás gyorsítása.

[1] Inline hívás + optimalizálás.

Strukturák visszaadása visszatérési értékként (optimalizált 1.)

```
00000000 _main        proc near
00000000 var_80
00000000
00000001
00000003
00000006
0000000C
0000000E                                    c_init
00000013                                    50]
00000018                                    +var_80], 8000h
00000020                                    sp+84h+var_80]
00000025              push    Eh[1]
00000027              push    14h
00000029              push    5
0000002B              push    offset ??_C@_0L@A@?$CJ?$AA@ ; "(%d,%d,%d)"
00000030              call    _printf
00000035              xor     eax, eax
00000037              mov     esp, ebp
00000039              pop     ebp
0000003A              retn
0000003A _main        endp
```

A fordító tudja, hogy az init_A() függvény mit csinál, ezért tudja, hogy milyen értékeket kell átadni a struktúramezők kiíratásához. Az init_A() függvényt nem hívja meg, és az r struktúrát sem inicializálja.

[1] Inline hívás + optimalizálás.

Strukturák visszaadása visszatérési értékként (optimalizált 2.)

```
00000050 _init_A    proc near
00000050 arg_0      = dword ptr  8
00000050 arg_4      = dword ptr  0Ch
00000050            push    edi
00000051            mov     ecx, 14h
00000056            mov     eax, [esp+arg_0]
0000005A            mov     edi, 1Eh
0000005F            mov     edx, [esp+arg_4]
00000063            mov     [eax], edx
00000065            mov     [eax+4], ecx
00000068            mov     [eax+8], edi
0000006B            pop     edi
0000006C            retn
0000006C _init_A            endp
```

Section 3

Objektumorientált kód

C++ függvénynév túlterhelés

A C++ fordító a tradicionális linkert használja, ezért az azonos nevű, de különböző paraméterű függvényeket át kell nevezni. Ezt name mangling-nak nevezzük. A különböző fordítók más-más elnevezési szabályt alkalmaznak.

```
int  f (void) { return 1; }
int  f (int)  { return 0; }
void g (void) { int i = f(), j = f(0); }
extern "C" void h(int i ) { f(i); }
```

g++

```
int  __Z1fv (void) { return 1; }
int  __Z1fi (int)  { return 0; }
void __Z1gv (void) { int i = __Z1fv(), j = __Z1fi(0); }
void _h(int i ) { __Z1fi(i); }
```

Visual C++

```
int  ?f@@YAHXZ (void) { return 1; }
int  ?f@@YAHH@Z (int)  { return 0; }
void ?g@@YAXXZ (void) { int i = ?f@@YAHXZ(), j = ?f@@YAHH@Z(0); }
void _h(int i ) { ?f@@YAHH@Z(i); }
```

C++ objektum modell

```
class Point {
public :
   Point( int x );
   virtual ~Point();
   int X() const;
   static int Count() ;
   virtual void Print() const;

protected :
   int x;
   static int _count;
} ;

Point::Point( int x ) : x(x)  {  _count++; }
Point::~Point() { _count--; }
int Point::X() const { return x; }
int Point::Count() { return _count; }
void Point::Print() const { printf("%d ", x); }

int Point::_count = 0;

int main() {
   int n;
   Point p1(4);
   n = Point::Count();  // stat fv. hívás demo
   return 0;
}
```

Objektumorientált program (main)

```
00000030 _main      proc near
00000030 var_14     = dword ptr -14h
00000030 var_C      = dword ptr -0Ch
00000030 var_8      = dword ptr -8
00000030 var_4      = dword ptr -4
00000030            push    ebp
00000031            mov     ebp, esp
00000033            sub     esp, 3
00000036            and     esp, 0FFFFFFF8h
00000039            add     esp, 4
0000003C            sub     esp, 18h
0000003F            lea     eax, [ebp+var_14]
00000042            mov     dword ptr [esp], 4
00000049            mov     ecx, eax
0000004B            call    ??0Point@@QAE@H@Z ; Point::Point(int)
00000050            mov     [ebp+var_C], eax
00000053            call    ?Count@Point@@SAHXZ ; Point::Count(void)
00000058            mov     [ebp+var_8], eax
0000005B            mov     eax, [ebp+var_8]
0000005E            mov     [ebp+var_4], eax
00000061            lea     eax, [ebp+var_14]
00000064            mov     ecx, eax
00000066            call    ??1Point@@UAE@XZ ; Point::~Point(void)
0000006B            mov     eax, 0
00000070            leave
00000071            retn
00000071 _main      endp
```

Objektumorientált program (main)

```
00000030 _main      proc near
00000030 var_14     = dword ptr -14h
00000030 var_C      = dword ptr -0Ch
00000030 var_8      = dword ptr -8
00000030 var_4      = dword ptr -4
00000030            push
00000031            mov
00000033            sub
00000036            and
00000039            add
0000003C            sub         esp, 8h
0000003F            lea         eax, [ebp+var_14]
00000042            mov         dword ptr [esp], 4
00000049            mov         ecx, eax
0000004B            call        ??0Point@@QAE@H@Z ; Point::Point(int)
00000050            mov         [ebp+var_C], eax
00000053            call        ?Count@Point@@SAHXZ ; Point::Count(void)
00000058            mov         [ebp+var_8], eax
0000005B            mov         eax, [ebp+var_8]
0000005E            mov         [ebp+var_4], eax
00000061            lea         eax, [ebp+var_14]
00000064            mov         ecx, eax
00000066            call        ??1Point@@UAE@XZ ; Point::~Point(void)
0000006B            mov         eax, 0
00000070            leave
00000071            retn
00000071 _main      endp
```

p1 objektum címe (this pointer) az **eax**-be.

Objektumorientált program (main)

```
00000030 _main      proc near
00000030 var_14     = dword ptr -14h
00000030 var_C      = dword ptr -0Ch
00000030 var_8      = dword ptr -8
00000030 var_4      = dword ptr -4
00000030            push    ebp
00000031            mov     ebp, esp
00000033            sub
00000036                                FFFF8h
00000039
0000003C
0000003F            lea             x, [ebp+var_14]
00000042            mov             word ptr [esp], 4
00000049            mov     ecx, eax
0000004B            call    ??0Point@@QAE@H@Z ; Point::Point(int)
00000050            mov     [ebp+var_C], eax
00000053            call    ?Count@Point@@SAHXZ ; Point::Count(void)
00000058            mov     [ebp+var_8], eax
0000005B            mov     eax, [ebp+var_8]
0000005E            mov     [ebp+var_4], eax
00000061            lea     eax, [ebp+var_14]
00000064            mov     ecx, eax
00000066            call    ??1Point@@UAE@XZ ; Point::~Point(void)
0000006B            mov     eax, 0
00000070            leave
00000071            retn
00000071 _main              endp
```

p1 objektum címe (this pointer) az **ecx**-be.

Objektumorientált program (main)

```
00000030 _main      proc near
00000030 var_14     = dword ptr -14h
00000030 var_C      = dword ptr -0Ch
00000030 var_8      = dword ptr -8
00000030 var_4      = dword ptr -4
00000030            push    ebp
00000031            mov     ebp, esp
00000033            sub     esp, 3
00000036            and     esp, 0FFFFFFF8h
00000039            add     esp, 4
0000003C
0000003F                                    var_14]
00000042            mov                 ptr [esp], 4
00000049            mov     cx, eax
0000004B            call    ?0Point@@QAE@H@Z ; Point::Point(int)
00000050            mov     [ebp+var_C], eax
00000053            call    ?Count@Point@@SAHXZ ; Point::Count(void)
00000058            mov     [ebp+var_8], eax
0000005B            mov     eax, [ebp+var_8]
0000005E            mov     [ebp+var_4], eax
00000061            lea     eax, [ebp+var_14]
00000064            mov     ecx, eax
00000066            call    ??1Point@@UAE@XZ ; Point::~Point(void)
0000006B            mov     eax, 0
00000070            leave
00000071            retn
00000071 _main              endp
```

Konstruktor hívás.

Objektumorientált program (main)

```
00000030 _main      proc near
00000030 var_14     = dword ptr -14h
00000030 var_C      = dword ptr -0Ch
00000030 var_8      = dword ptr -8
00000030 var_4      = dword ptr -4
00000030            push    ebp
00000031            mov     ebp, esp
00000033            sub     esp, 3
00000036            and     esp, 0FFFFFFF8h
00000039            add     esp, 4
0000003C            sub     esp, 18h
0000003F            lea     eax, [ebp+var_14]
00000042            mov     dword ptr [esp], 4
00000049            mov     ecx, eax
0000004B            call    ??0Point@@QAE@H@Z ; Point::Point(int)
00000050            mov     [ebp+var_C], eax
00000053                    t@@SAHXZ ; Point::Count(void)
00000058                    ], eax
0000005B            mov     x, [ebp+var_8]
0000005E            mov     ebp+var_4], eax
00000061            lea     eax, [ebp+var_14]
00000064            mov     ecx, eax
00000066            call    ??1Point@@UAE@XZ ; Point::~Point(void)
0000006B            mov     eax, 0
00000070            leave
00000071            retn
00000071 _main      endp
```

Destruktor hívás (this az **ecx**-ben).

Objektumorientált program (main)

```
00000030 _main      proc near
00000030 var_14     = dword ptr -14h
00000030 var_C      = dword ptr -0Ch
00000030 var_8      = dword ptr -8
00000030 var_4      = dword ptr -4
00000030            push    ebp
00000031            mov     ebp, esp
00000033            sub     esp, 3
00000036            and     esp, 0FFFFFFF8h
00000039            add     esp, 4
0000003C            sub     esp, 18h
0000003F                    var_14]
00000042                    esp], 4
00000049
0000004B            call    oint@@QAE@H@Z ; Point::Point(int)
00000050            mov     [ebp+var_C], eax
00000053            call    ?Count@Point@@SAHXZ ; Point::Count(void)
00000058            mov     [ebp+var_8], eax
0000005B            mov     eax, [ebp+var_8]
0000005E            mov     [ebp+var_4], eax
00000061            lea     eax, [ebp+var_14]
00000064            mov     ecx, eax
00000066            call    ??1Point@@UAE@XZ ; Point::~Point(void)
0000006B            mov     eax, 0
00000070            leave
00000071            retn
00000071 _main      endp
```

Static tagfüggvény, this nem kerül átadásra.

Objektumorientált program (konstruktor Microsoft C)

```
00000072 ??0Point@@QAE@H@Z proc near
00000072 var_4 = dword ptr -4
00000072 arg_0 = dword ptr  8
00000072    push   ebp
00000073    mov    ebp, esp
00000075    push   esi
00000076    mov    [ebp+var_4], ecx
00000079    mov    eax, [ebp+var_4]
0000007C    mov    dword ptr [eax],offset ??_7Point@@6B@ ;Point::vftable
00000082    mov    eax, [ebp+var_4]
00000085    mov    edx, [ebp+arg_0]
00000088    mov    [eax+4], edx
0000008B    mov    eax, 1
00000090    add    eax, ds:?_count@Point@@1HA ; int Point::_count
00000096    mov    ds:?_count@Point@@1HA, eax ; int Point::_count
0000009B    mov    eax, [ebp+var_4]
0000009E    leave
0000009F    retn     4
0000009F ??0Point@@QAE@H@Z endp
```

Objektumorientált program (konstruktor Microsoft C)

```
00000072 ??0Point@@QAE@H@Z proc near
00000072 var_4 = dword ptr -4
00000072                    8
00000072
00000073    mov     p, esp
00000075    push    si
00000076    mov     [ebp+var_4], ecx
00000079    mov     eax, [ebp+var_4]
0000007C    mov     dword ptr [eax],offset ??_7Point@@6B@ ;Point::vftable
00000082    mov     eax, [ebp+var_4]
00000085    mov     edx, [ebp+arg_0]
00000088    mov     [eax+4], edx
0000008B    mov     eax, 1
00000090    add     eax, ds:?_count@Point@@1HA ; int Point::_count
00000096    mov     ds:?_count@Point@@1HA, eax ; int Point::_count
0000009B    mov     eax, [ebp+var_4]
0000009E    leave
0000009F    retn    4
0000009F ??0Point@@QAE@H@Z endp
```

this pointer elmentése.

Objektumorientált program (konstruktor Microsoft C)

```
00000072 ??0Point@@QAE@H@Z proc near
00000072 var_4 = dword ptr -4
00000072 arg_0 = dword ptr  8
00000072
00000073
00000075
00000076    mov     [ebp+var_4], ecx
00000079    mov     eax, [ebp+var_4]
0000007C    mov     dword ptr [eax], offset ??_7Point@@6B@ ; Point::vftable
00000082    mov     eax, [ebp+var_4]
00000085    mov     edx, [ebp+arg_0]
00000088    mov     [eax+4], edx
0000008B    mov     eax, 1
00000090    add     eax, ds:?_count@Point@@1HA ; int Point::_count
00000096    mov     ds:?_count@Point@@1HA, eax ; int Point::_count
0000009B    mov     eax, [ebp+var_4]
0000009E    leave
0000009F    retn    4
0000009F ??0Point@@QAE@H@Z endp
```

virtuális függvénytábla beállítása.

Objektumorientált program (konstruktor Microsoft C)

```
00000072 ??0Point@@QAE@H@Z proc near
00000072 var_4 = dword ptr -4
00000072 arg_0 = dword ptr  8
00000072    push   ebp
00000073    mov    ebp, esp
00000075    push   esi
00000076                  ], ecx
00000079                  var_4]
0000007C    mov           ptr [eax], offset ??_7Point@@6B@ ; Point::vftable
00000082    mov    eax, [ebp+var_4]
00000085    mov    edx, [ebp+arg_0]
00000088    mov    [eax+4], edx
0000008B    mov    eax, 1
00000090    add    eax, ds:?_count@Point@@1HA ; int Point::_count
00000096    mov    ds:?_count@Point@@1HA, eax ; int Point::_count
0000009B    mov    eax, [ebp+var_4]
0000009E    leave
0000009F    retn     4
0000009F ??0Point@@QAE@H@Z endp
```

x tagváltozó inicializálása.

Objektumorientált program (konstruktor Microsoft C)

```
00000072 ??0Point@@QAE@H@Z proc near
00000072 var_4 = dword ptr -4
00000072 arg_0 = dword ptr  8
00000072    push    ebp
00000073
00000075
00000076                                    ], ecx
00000079                                    var_4]
0000007C                                    [eax], offset ??_7Point@@6B@ ; Point::vftable
00000082                                    bp+var_4]
00000085    mov     x, [ebp+arg_0]
00000088    mov     eax+4], edx
0000008B    mov     eax, 1
00000090    add     eax, ds:?_count@Point@@1HA ; int Point::_count
00000096    mov     ds:?_count@Point@@1HA, eax ; int Point::_count
0000009B    mov     eax, [ebp+var_4]
0000009E    leave
0000009F    retn    4
0000009F ??0Point@@QAE@H@Z endp
```

_counter növelése. Statikus, ezért a default adatsegmensben található.

Lehetséges C++ objektum modell