

Öğrenci Adı – Soyadı: _____
Öğrenci Numarası: _____

<i>S1</i>	<i>S2</i>	<i>S3</i>	<i>S4</i>	<i>S5</i>	<i>Toplam</i>



Hacettepe Üniversitesi
Bilgisayar Mühendisliği Bölümü

BBM 341
Sistem
Programlama
Ara Sınav

Tarih: 19 Kasım 2012

Süre: 105 dak.

Sınavı başılamadan önce aşağıda yazılanları mutlaka okuyunuz!

- Bu sınav **kapalı kaynak** bir sınavdır. Yani sınav süresince ilgili ders kitapları veya ders notlarınızdan faydalanmanız yasaktır.
- Size yardımcı olması açısından sonraki iki sayfada bazı Intel IA32/x86-64 Assembly komutlarının söz dizimleri ve diğer bazı ilgili tanımlar verilmiştir.
- **Sınavda kopya çekmek yasaktır.** Kopya çekmeye teşebbüs edenler hakkında ilgili idare işlemler **kesinlikle** başlatılacaktır.
- Her bir sorunun toplam ağırlığı soru numarasının ardında parantez içinde belirtilmiştir.
- Sınav toplam 100 puan üzerinden değerlendirilecektir.

Sınav bu kapak sayfası dahil toplam 9 sayfadan oluşmaktadır. Lütfen kontrol ediniz!

BAŞARILAR!

Sıçrama İşlemleri

Sıçrama	Koşul
jmp	1
je	ZF
jne	~ZF
js	SF
jns	~SF
jg	~(SF^OF) & ~ZF
jge	~(SF^OF)
jl	(SF^OF)
jle	(SF^OF) ZF
ja	~CF & ~ZF
jb	CF

Aritmetik İşlemler

Format	İşlem
addl <i>Src, Dest</i>	Dest = Dest + Src
subl <i>Src, Dest</i>	Dest = Dest - Src
imull <i>Src, Dest</i>	Dest = Dest * Src
sall <i>Src, Dest</i>	Dest = Dest << Src
sarl <i>Src, Dest</i>	Dest = Dest >> Src
shrl <i>Src, Dest</i>	Dest = Dest >> Src
xorl <i>Src, Dest</i>	Dest = Dest ^ Src
andl <i>Src, Dest</i>	Dest = Dest & Src
orl <i>Src, Dest</i>	Dest = Dest Src
incl <i>Src</i>	Dest = Dest + 1
decl <i>Src</i>	Dest = Dest - 1
negl <i>Src</i>	Dest = - Dest
notl <i>Src</i>	Dest = ~ Dest

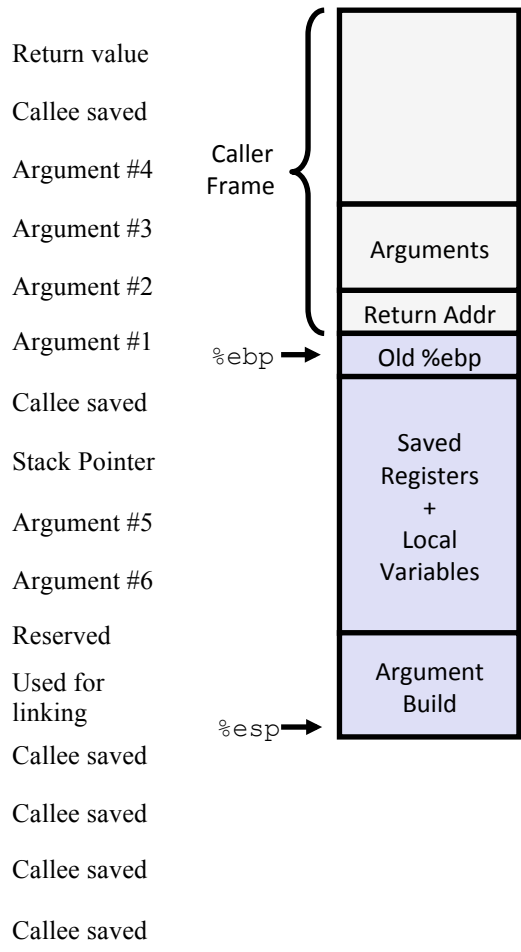
Bellek İşlemleri

Format	İşlem
(Rb, Ri)	Mem[Reg[Rb]+Reg[Ri]]
D(Rb, Ri)	Mem[Reg[Rb]+Reg[Ri]+D]
(Rb, Ri, S)	Mem[Reg[Rb]+S*Reg[Ri]]

Yazmaçlar (Registers)

63	31	15	8	7	0
%rax	%eax %ax	%ah	%al		
%rbx	%ebx %bx	%bh	%bl		
%rcx	%ecx %cx	%ch	%cl		
%rdx	%edx %dx	%dh	%dl		
%rsi	%esi %si		%sil		
%rdi	%edi %di		%dil		
%rbp	%ebp %bp		%bpl		
%rsp	%esp %sp		%spl		
%r8	%r8d %r8w		%r8b		
%r9	%r9d %r9w		%r9b		
%r10	%r10d %r10w		%r10b		
%r11	%r11d %r11w		%r11b		
%r12	%r12d %r12w		%r12b		
%r13	%r13d %r13w		%r13b		
%r14	%r14d %r14w		%r14b		
%r15	%r15d %r15w		%r15b		

Linux Yığıt (Stack) Yapısı



Özel Hizalama Durumları (Intel IA32)

- 1 byte: `char`, ...
sınırlandırma yok
- 2 bytes: `short`, ...
en düşük bit adresi 0_2
- 4 bytes: `int`, `float`, `char *`, ...
en düşük 2 bit adresi 00_2
- 8 bytes: `double`, ...
Windows: *en düşük 3 bit adresi 000_2*
Linux: *en düşük 2 bit adresi 00_2*
- 12 bytes: `long double`
Windows & Linux:
en düşük 2 bit adresi 00_2

C Veri Tipi	IA-32	X86-64
<code>char</code>	1	1
<code>short</code>	2	2
<code>int</code>	4	4
<code>long</code>	4	8
<code>long long</code>	8	8
<code>float</code>	4	4
<code>double</code>	8	8
<code>long double</code>	10/12	10/16
<code>pointer</code>	4	8

Özel Hizalama Durumları (Intel x86-64)

- 1 byte: `char`, ...
sınırlandırma yok
- 2 bytes: `short`, ...
en düşük bit adresi 0_2
- 4 bytes: `int`, `float`, ...
en düşük 2 bit adresi 00_2
- 8 bytes: `double`, `char *`, ...
Windows & Linux:
en düşük 3 bit adresi 000_2
- 16 bytes: `long double`
Linux: *en düşük 3 bit adresi 000_2*

Bayt Sıralama (Byte Ordering)

$0x100$ adresinde 4-baytlık değişken
 $0x01234567$

Big Endian

En anlamsız bayt en yüksek adreste

$0x100$ $0x101$ $0x102$ $0x103$

01	23	45	67
----	----	----	----

Little Endian

En anlamsız bayt en düşük adreste

$0x100$ $0x101$ $0x102$ $0x103$

67	45	23	01
----	----	----	----

Kayan noktalı sayı (floating point)

$$\text{Bias} = 2^{k-1} - 1$$

Soru 1. (17 puan) Tamsayı gösterimleri.

6-bit ve 4-bit'lik iki bilgisayar sistemi (M1 ve M2) üzerinde,

- *İşaretili tam sayılar (signed integers) için ikili tümler (2's complement) aritmetiği* kullanılmaktadır.
- `short` tamsayılar M1'de 3-bit, M2'de ise toplam 2-bit ile gösterilmektedir.
- Bir `short` açıkça `int`'e dönüştürülürken (*cast* edilirken) *işaret genişletmesi (sign extension)* kendiliğinden gerçekleşmektedir.
- `int`'ler üzerinde sağa kaydırma *aritmetik kaydırma (arithmetic shift)* işlemi ile gerçekleşmektedir.

Bu varsayımlara göre aşağıdaki tanımları göz önünde bulundurarak altta verilen tablodaki boş kutucukları doldurunuz.

NOT: “-” ile belirtilen bölümleri doldurmanıza gerek yoktur.

```
int a = 2;  
int b = -2*a;  
short sa = (short) b;  
unsigned ub = b;
```

İfade	M1 sisteminde tam sayı gösterimi	M1 sisteminde ikili gösterimi	M2 sisteminde tam sayı gösterimi	M2 sisteminde ikili gösterimi
-	19		-	-
-	-	-		11 01
b				
sa				
ub				
a << 2				
b >> 1				
<i>Tmax</i>				
a b				
a ⁻¹				

Soru 2. (20 puan) Kayan noktalı sayı gösterimleri.

Bu soruyu IEEE Standard 754 kayan noktalı sayı formatına göre oluşturulan 8-bit'lik bir kayan noktalı gösterimine göre cevaplayınız. Bu gösterimde,

- En anlamlı bit (the most significant bit) *işaret bit'*idir.
- İşaret bit'inin ardından gelen 3 bit kayan noktalı sayının *üstünü* (*exponent*) verir.
- Geri kalan 4 bit ise *kesirli kısmı* (*fraction*) belirtir.

NOT: Aşağıdaki soruları cevaplarken hem kesirli gerçek değerleri hem de ikili gösterimleri belirtiniz.

(a) (2 puan) Bu gösterimde üst için kaydırma değeri (exponent bias) nedir?

(b) (2 puan) Bu gösterimde ifade edilen en küçük pozitif sayı nedir?

(c) (2 puan) Bu gösterimde normalize olmayan en büyük pozitif sayı nedir?

(d) (2 puan) Bu gösterimde normalize olan en küçük pozitif sayı nedir?

(e) (2 puan) Bu gösterimde pozitif sonsuz nasıl ifade edilmektedir?

(f) (5 puan) Aşağıda verilen tablodaki boş kutucukları doldurunuz. Eğer bilgisayardaki gösterimde yuvarlama gerekiyor ise çifte-yuvarlama (round-to-even) yapmalısınız

İkili gösterim	Ondalık Değer
0 000 0000	0
1 110 0001	
	-11
	1/4
1 000 1101	
	23/256

Soru 3. (24 puan) Assembly/C çevrimi.

Aşağıda bir C fonksiyonu için derleyici tarafından üretilen Assembly kodu gösterilmektedir:

```
08048380 <foo>:
8048380:    push   %ebp
8048381:    mov    %esp,%ebp
8048383:    mov    0x8(%ebp),%edx
8048386:    cmp    $0x40,%edx
8048389:    ja    80483b1 <foo+0x31>
804838b:    mov    $0x20,%ecx
8048390:    test   $0x3,%dl
8048393:    jne    804839c <foo+0x1c>
8048395:    jmp    80483b1 <foo+0x31>
8048397:    test   $0x3,%dl
804839a:    je    80483b6 <foo+0x36>
804839c:    mov    %ecx,%eax
804839e:    shr   $0x1f,%eax
80483a1:    add    %ecx,%eax
80483a3:    sar   %eax
80483a5:    lea   0x1(%eax),%ecx
80483a8:    add    %edx,%edx
80483aa:    cmp    $0x40,%edx
80483ad:    jbe    8048397 <foo+0x17>
80483af:    jmp    80483b6 <foo+0x36>
80483b1:    mov    $0x20,%ecx
80483b6:    mov    %ecx,%eax
80483b8:    pop   %ebp
80483b9:    ret
```

Yukarıda verilen Assembly koduna göre aşağıdaki C kodunda yer alan boşlukları doldurunuz.

```
unsigned foo(unsigned x)
{
    int y, z;

    for(z = _____; _____; _____) {
        y = _____;
        if (y == 0) {
            _____;
        }
        _____;
    }
    return z;
}
```

Soru 4. (32 puan) Yöntemler ve yığıt.

Aşağıda iki C fonksiyonu ve ilgili Assembly kodları verilmektedir:

```
int fun1 (int x, int *y)
{
    return 24*x+*y;
}

int fun2(int x, int y)
{
    return fun1(y, &x);
}
```

```
08048374 <fun1>:
8048374:    push   %ebp
8048375:    mov    %esp,%ebp
8048377:    mov    0x8(%ebp),%eax
804837a:    lea   (%eax,%eax,2),%eax
804837d:    shl   $0x3,%eax
8048380:    mov    0xc(%ebp),%edx
8048383:    add   (%edx),%eax
8048385:    pop   %ebp
8048386:    ret

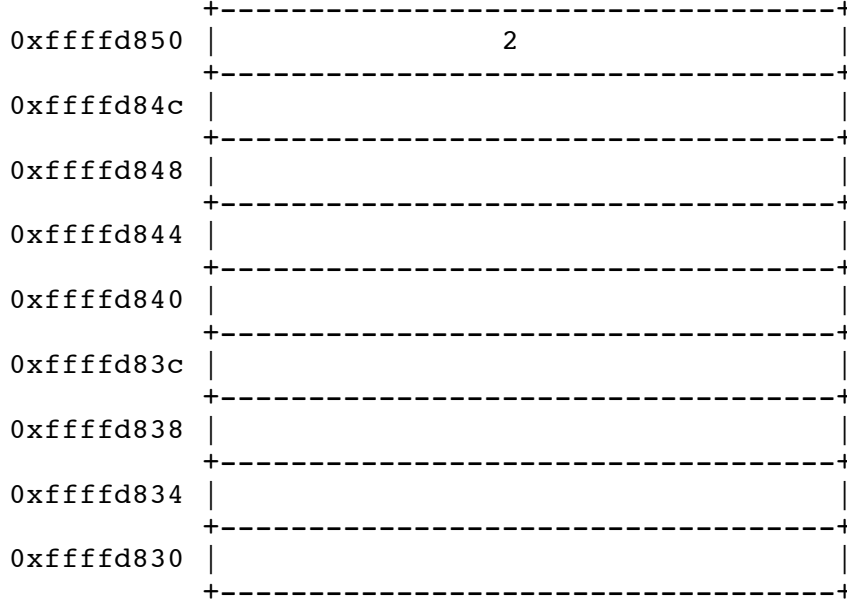
08048387 <fun2>:
8048387:    push   %ebp
8048388:    mov    %esp,%ebp
804838a:    sub   $0x8,%esp
804838d:    lea   0x8(%ebp),%eax
8048390:    mov    %eax,0x4(%esp)
8048394:    mov    0xc(%ebp),%eax
8048397:    mov    %eax,(%esp)
804839a:    call  8048374 <fun1>
804839f:    leave
80483a0:    ret
```

(a) (24 puan) fun2(1,2) fonksiyon çağrısından başlayarak fun1 fonksiyonundaki ret komutunun işletilmesinden hemen önceki yığıtın detaylı içeriğini aşağıdaki diyagram üzerinde belirtiniz. Bu soruyu çözerken,

- Yığıt diyagramı fun2 fonksiyonunun iki argümanı ile başlamalıdır, örnek olması açısından argümanlardan biri sizin yerinize yerleştirilmiştir.
- Yığıt içeriğini belirtirken değişken isimleri yerine gerçek değerleri kullanınız.
- %ebp yazmacının değerini yığıt üzerine yerleştirmeniz gerekiyorsa bu değer %ebp'ye ait olduğunu da açıkça belirtiniz, örneğin %ebp: 0xffff1400.
- Size sağlanan yığıt diyagramında tüm kutucukları kullanmanıza gerek olmayabilir.

fun2 cagrisi gerceklestirildigi anda %ebp degeri: 0xffffd858
fun2 cagrisindan geri donus adresi: 0x080483c9

Yigit Asagidaki diyagram fun2 fonksiyonuna gerceklestiren
adresleri bir cagrinin ardindaki durumu gostermektedir



(b) (4 puan) fun1 için ret komutunun çalıştırılmasının hemen öncesinde %ebp'nin değeri nedir?

%ebp: 0x _____

(c) (4 puan) fun1 için ret komutunun çalıştırılmasının hemen öncesinde %esp'nin değeri nedir?

%esp: 0x _____

Soru 5. (7 puan) *Okuma ödevleri.*

(a) (2 puan) IEEE-754 kayan noktalı sayı formatında sıfır sayısının +0 ve -0 olarak iki farklı temsilinin olması ne gibi bir avantaj sağlamaktadır? 1-2 cümle ile açıklayınız.

(b) (2 puan) Hangi iki Bell Labs çalışanı hem Unix işletim sisteminin hem de C programlama dilinin geliştirilmesinde aktif rol almışlardır? Soyadlarını yazmanız yeterlidir.

(c) (3 puan) Daha sonra Intel tarafından kısmen benimsenen ilk olarak AMD'nin geliştirdiği x86-64 komut kümesinin Intel'in bir başka 64-bit'lik mimarisi olan IA-64'den en temel farkı nedir?