Tölvutækni og forritun Lokapróf

brj46

Nóvember 2024

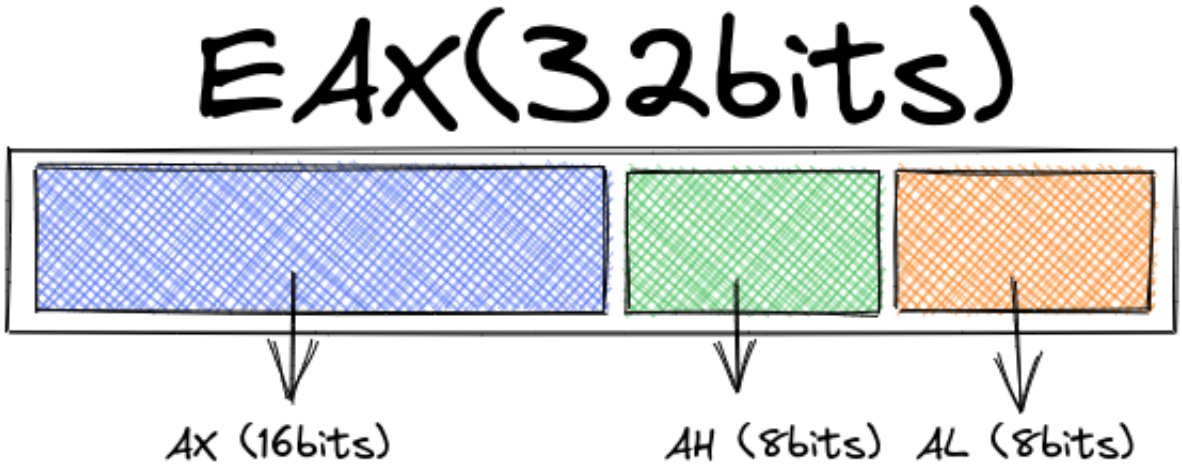
Hvað þarf ég að kunna fyrir prófið?

- ☐ Bitavinnsla með heiltölur (signed og unsigned)
- ☐ Einfaldar fleytitölur
- ☐ Skrifa C kóða út frá smalamálskóða (reikniaðgerðir, styriskipanir, föll, bestun,...)
- ☐ Minnisyfirflæði (buffer overflow)
- ☐ Skyndiminni (Skipulag og notkun)
- ☐ Frabrigði, ferlar (fork, wait)
- ☐ Skipulag á sýndarminni
- ☐ Minnisúthlutun

Tekið frá Tuma

assembly

Uppbrot Gistis



algengar skipanir

skipun	argument	lýsing
mov	x, y	færir úr x yfir í y, sjá conditional move fyrir neðan
push	x	ýtir x á hlaða og eftir að hækka %ESP um sizeof(x) bæti og sett þar inn
pop	x	skilar síðasta gildi sem var sett á hlaðann inn í x
lea	(x), y	lea, betur þekkt sem leaq er notað til að framkvæma reikning (x) og setja útkomu inn í y
(x,y)		skilar útkomu úr reikningi x + y
0,(,x,y)		skilar útkomu úr reikningi x * y
(x,y,z)		skilar útkomu úr reikningi x + y * z
2(x, y, z)		skilar útkomu úr reikningi (2 + (x + y * z))
sar	x, y	hliðrar y um x bita til hægri, basically heiltöludeiling með x
sal	x, y	næstum eins og sar nema til vinstri, núna með margföldun með 2^x
sub	x,y	dregur y frá x
inc/dec	x	hækkar/lækkar gildi x um 1

ath. **SHL** og **SAL** gera það sama en **SHR** virkar ekki með signed int eins og **SAR**

Algeng mynstur

mynstur	skýring
testl %edi, %edi	logical andað **edi** við **edi** þannig ef $edi <= 0$ er hægt að cmove eða jc í samræmi við það
cmove $5, %eax	færðu 5 inn í eax ef z-flaggið er sett sem 1 þ.e. ef edi er tómt
leal 0(%rdi, %rdi, 4)	margfaldar %rdi með 5, $(x + 4 * x)$

conditional codes

þessir kóðar fara a endann á cmov skipunum þ.e. cmov– í línu eftir að eitthvað er testað eins og í dæmi

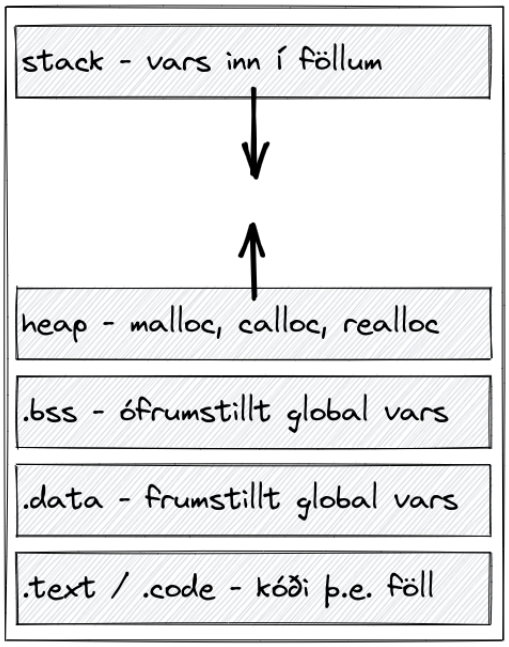
```
testb   $7, %dl
cmove   $1, %rax
```

Þetta er pínu fucked dæmi því e flaggið í cmove stendur fyrir equal nema hvað við erum actually að athuga hvort útkomugildið sé 0, þ.e. að enginn af neðstu 3 bitunum sé 1, þá er gott að muna að e er jafngilt z

þessi koði færir 1 inn í %rax ef neðstu þrír bitar %dl eru ekki 111

cc	condition
o	overflow
no	no overflow
b, nae	below, not above or equal
nb, ae	not below, above or equal
e, z	equal(zero)
ne, nz	not equal, (not zero)
na, be	not above, below or equal
a, nbe	above, not below or equal
s	sign
ns	no sign
p	parity
np	no parity
l, nge	less, not greater than or equal
nl	not less, greater than or equal
ng, le	not greater, less than or equal
g, nle	greater, not less than or equal

minnissvæði



ath. global breytur sem eru skilgreindar sem 0 eða NULL eru líka í .bss

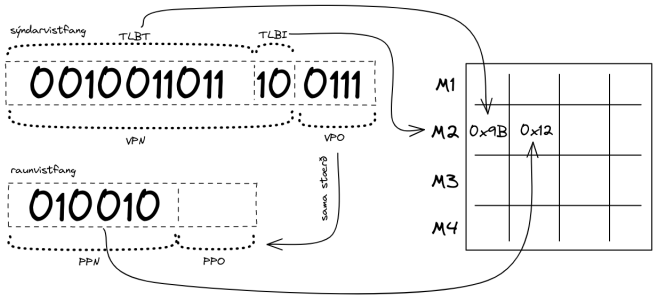
Sýndarminni

- sýndarvistföng: a bitar
- raunvistföng: b bitar
- síðustærð: c bæti
- TLB: d vítt, e sæti
- fjöldi mengha: f

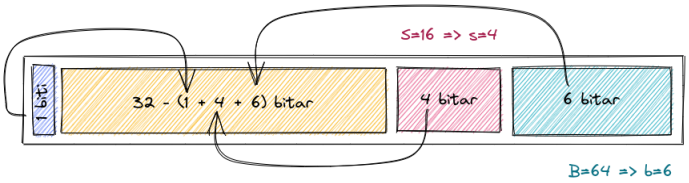
fjöldi mengja er reiknað $\frac{e}{d} = f$

við erum með sýndarvistfang sem er 16 bitar sem skiptast í 4 mengi þá er **VPN** $\frac{3}{4} \times 16$ bitar og **VPO** 4 bitar

TBLT og **TLBI** eru skipting á **VPN** og **TBLT** restin raunvistföngin eru jafn löng og **TLBT** og skipt niður í tvo hluta **PPN** og **PPO**, sem er jafn stór og **VPO**(í þessu tilfelli 4 bitar)



annap dæmi, við erum með sýndarminni sem er 4kb að stærð, 4-vítt, E, og með 16 mengi, S, svo útfrá þessum tölum finnum við línustærð, B, með reikningnum $\frac{4096}{16 \times 4} = 64$ skiptum þessu nu upp fyrir 32-bita vistfang:



klukkutifsformúla

$a + s \times r = m$

- aðgangstími = a(tif)
- smellahlutfall = s(hlutfall)
- smellarefsing = r (tif)
- meðalaðgangstími = m (tif)

dæmi:

- 97% smellahlutfall, $1 + 0.03 \times 100 = 4$
- 99% smellahlutfall, $1 + 0.01 \times 100 = 2$

Próf 2022

1

Í þessu dæmi ætlum við að nota unsigned long breytur til að tákna (allt að) 64 staka mengi (sets). Ef a er unsigned long breyta þá er stak i í menginu a ef biti i ($i = 0, \ldots 63$) er 1, annars er stak i ekki í menginu. Til dæmis væri mengið $\{0, 3\}$, táknað með 64-bita bitastrengnum 00...01001. Athugið að bitarnir eru númeraðir frá hægri til vinstri, svo stak 0 er í menginu, en stak 1 er ekki í menginu, stak 2 er ekki í menginu, o.s.frv.

a.

Skrifið einnar línu fall (þ.e. bara ein **return** skipun) sem skilar sammengi (union) tveggja slíkra mengja. Haus fallsins: **unsigned long sammengi(unsigned long a, unsigned long b)**

Svar:

```
unsigned long sammengi(unsigned long a, unsigned long b){
    return a | b;
}
```

b.

Skrifið einnar línu fall sem skilar mengjamun (set difference) mengjanna a og b, þ.e. öll stök sem eru í a, en ekki í b. Haus fallsins: **unsigned long munur(unsigned long a, unsigned long b)**

Svar:

```
unsigned long munur(unsigned long a, unsigned long b){
    return a & ~b;
}
```

c.

Athugið að í C er fastinn **1ul** (tölustafurinn **1** og bókstafirnir u og l) 64-bita heiltalan 1 án formerkis. Hvaða mengi táknar segðin **(**$1ul << i$**)**?

Svar: Segðin **(**$1ul << i$**)** táknar mengið sem inniheldur stakið i og engin önnur stök.

d.

Skrifið einnar línu fall sem skilar því hvort stak i sé í menginu a. Skilagildið á að vera 1 (*satt*) ef i er í a, en 0 (*ósatt*) annars. Þið megið gera ráð fyrir því að gildið á i sé á bilinu 0 til 63. Haus fallsins: **int stakI(unsigned long a, int i)**

Svar:

```
    int stakI(unsigned long a, int i){
        return (a >> i) & 1;
    }
```

2

Við höfum 10-bita fleytitölur sem fylgja IEEE staðlinum. Við vitum ekki skiptingu þeirra í veldishluta (exp) og brothluta (frac), en það er einn formerkisbiti fremst í þeim.

Vika 1

Kynning, Linux, C

Heimadæmi spurningar

1 og 2

kennslu aðferðir

3

Skoðið sýnidæmin á glæru 16 í fyrirlestri 1 (þ.e. $50000 * 50000$ fyrir int og $1e20 + (-1e20 + 3.14)$ fyrir float).

a . Skrifið stutt forrit í Java sem prentar út niðurstöðuna úr þessum útreikningunum (athugið að í Java eru kommutölufastar sjálfkrafa af taginu double. Til að fá float-fasta þarf að setja f á eftir fastanum, t.d. 3.14f).

b . Reyndar er gildið 1e20 (þ.e. 1020) óþarflega stórt. Það eru til mun minni gildi sem valda sömu vandræðum. Finnið lægsta gildi a á 10a sem gefur sömu niðurstöðu og 1e20 í seinni formúlunni á glæru 16. Sýnið útprentun á því í Java forriti.

4

Á glærum 20 og 21 í fyrirlestri 1 er sýnd minnisvilla sem getur komið upp í C forriti. Útskýrið hvers vegna svona villa myndi ekki koma upp í sambærilegu Java forriti. Hvaða kostir og gallar eru við það að leyfa möguleika á svona villum í C forritum?

5

Setjið upp linux

Vika 2

C, Bendar, minni, notkun

Vika 3

Upplýsingar sem bitar, heiltölur

Vika 4

Bætaröð, fleytitölur

Vika 5

Skipulag örgjava, smalamálsforritun

Vika6

Stýriskipanir og stef í smalamáli

Vika 7

Gögn og yfirflæði minnis

Vika 8

Bestun smalamálskóða

Vika 9

Minnisstigveldi, skyndiminni

Vika 10

Tenging, keyrsluskrár, forritasöfn

Vika 11

Frábrigði, ferlastýring

Vika 12

Sýndarminni

Vika 13

Minnisúthlutun, ruslasöfnun, minnisvillur

Vika 14

Samantekt