**Við skilgreinum 10-bita fleytitölur sem hegða sér eins og IEEE fleytitölurnar. Fremsti bitinn er formerkisbiti, síðan eru 5 bitar fyrir veldishluta (exp) og loks 4 bitar fyrir brothlutann (frac). Hvernig verður talan 2.5 táknuð með 10 bitum á þessu formi?**

2.5 er 10.1 í binary, færum kommu til vinstri um einn og fáum 1.01, þá er E = 1. Bias er 2 (5 veldisbitar – 1)- 1,eða 24-1 = 15. E er 1, svo 15+1 = 16, það er 10000 í binary. Finnum nú M höfum 01 eftir kommu svo fyllum við uppí 4 bita með 0 eða 0100 þá eru við kominn með 0-10000-0100.

**Notum fleytitöluna 0-10000-0100. Hver er stærsta tala sem hægt er að tákna sem staðlaða fleytitölu?**

Þá breytiru öllum bitum í veldishluta sem eru 0 í 1 nema síðasta, þá er það 0-11110-1111

M= 1+1/2 +1/4 + 1/8 +1/16 = 1.9375

Bíasið fyrir 5 veldishluta er 25-1-1 = 15

Veldishlutinn í tuga er 30-15 = 15

30 kemur frá 11110.

V = (-1)s \*M\*2E = (-1)x 1.9375 x 215 = 63552

**Í tölvu er skyndiminni (cache) sem er 16KB af stærð, skyndiminnið er 8-vítt og blokkastærð er 64 bæti. Hve mörg mengi eru í skyndiminninu?**

Skyndiminni = S\*B\*E

Skyndiminni = Mengi\*Blokkastærð\*Vítt

Mengi = (Skyndiminni)/(Blokkastærð\*Vítt)

Mengi = (16KB)/(64bæti\*8) = 32

**Ef við stækkum (RAM) í tölvu úr 8GB í 16GB, hvert af eftirfarandi mun aldrei breytast? (A) raunvistfang (physical address space), (B) afköst tölvunnar, (C) sýndar vistfangsrými (virtual address space), (D) fjöldi síða (page) sem eru í rauninni**

(C) = Sýndarvistfangsrými (virtual address space).

**Gerum ráð fyrir að við séum með 6-bita tölvu þar sem int er 6-bitar og short eru 3-bitar. Ef við keyrum eftirfarandi C kóða:**

**short sy = -2;**

**int y = sy;**

**int x = -15;**

**unsigned ux = x;**

**Hver eru gildin á eftirfarandi segðum:**

**(a) y, (b) ux, (c) x >> 1, (d) y-ux**

(a) short sy er -2 eða 110 fyrir 3 bita

Int y er 6 bitar sem er 111110 sem er -2

Y = -2

(b) ux = -15 er 100001 er gerum það svo unsigned

Eða 32+1 = 33.

(c) x >> 1 = hliðrað um 1 eða breytir 110000 = -16

(d) y-ux = -2-33 = -35

**Fyrir eftirfarandi fall:**

**int f(int x) {**

**int mask = x>>31;**

**return (x ^ mask) + ~mask + 1;**

**}**

**(a) Rekið ykkur í gegnum hvernig fallið vinnur fyrir inntökin -1 og 3.**

**(b) Útskýrið hvað fallið gerir og hvaða hlutverki mask breytan gegnir?**

Inntak: x = -1

Skref 1: mask = x >> 31;

Í 32-bita heiltölu er -1 framsett sem 11111111111111111111111111111111 (allir bitar eru 1). Bitafærsla til hægri um 31 bita mun hafa í för með sér að mask verður 11111111111111111111111111111111.

Skref 2: return (x ^ mask) + ~mask + 1;

x ^ mask er -1 ^ 11111111111111111111111111111111, sem er 0.

~mask er ~11111111111111111111111111111111, sem er 0.

~mask + 1 er því 1.

Lokaniðurstaða: 0 + 1 = 1.

Inntak: x = 3

Skref 1: mask = x >> 31;

Í 32-bita heiltölu er 3 framsett sem 00000000000000000000000000000011. Bitafærsla til hægri um 31 bita mun hafa í för með sér að mask verður 00000000000000000000000000000000.

Skref 2: return (x ^ mask) + ~mask + 1;

x ^ mask er 3 ^ 00000000000000000000000000000000, sem er 3.

~mask er ~00000000000000000000000000000000, sem er 11111111111111111111111111111111.

~mask + 1 er því 0.

Lokaniðurstaða: 3 + 0 = 3.

(b) Fallið er að finna two complement fyrir neikvæðar tölur og skilar jákvæðum óbreyttum, þetta er í raun að finna algildi inntakstölunnar. Mask breytan er notuð til að ákvarða hvort inntak x sé neikvætt eða jákvætt, það er gert með bitafærslu til hægri. Skilar algildi á X.

**Eftirfarandi ferlanet er gefið sem stafirnir A-E eru prentaðir með printf. Hvert af eftirfarandi prentast ekki út? (A) ADBCE, (B) ACBED, (C) ABCED, (D) ABEDC.**

**A diagram of a diagram

Description automatically generated**

**Hér fyrir fyrir neðan er x86-64 smalamálsútgáfa af endukvæmna fallinu unsigned long func (unsigned long n): (a) Sýnið jafngilt endurkvæmt C fall. (b) Teiknið upp hlaðramma (stack frame ) fyrir fallið func þegar kallað er á það með gildinu 13. (c) hvað gerir func?**

**A black screen with white text and colorful text

Description automatically generatedA screenshot of a black background with white text

Description automatically generated**

**A screen shot of a computer code

Description automatically generated**

**(a) Teiknið upp skipulag þessarar færslu í minni miðað við uppröðunarkröfu x86-64 linux.**

**(b) Segjum að gistið %rdi bendi fremst á færslu af þessari gerð og að gistið %rsi innihaldi gildið 2. Útskýrið hvað eftirfarandi smalamálsskipun mun gera, munið að %al gistið er neðsta bætið af %rax**

**movzbl 40(%rdi), %eax**

**movb %al, 24(%rdi, %rsi)**

**(c) Umraðið sviðunum þannig að færslan taki minna pláss. Rökstyðjið.**

**Gefið er tölvukerfi með 4-vítt skyndiminni, 8 mengi og blokkastærð 64 bæti. Gerum ráð fyrir að vistfangsrýmið er 1MB (220 bæti).**

**(a) í hvaða mengi varpast vistfangið 0x12345?**

4 vítt skyndiminni þá er fjöldi blokka 24=16 blokkir, fjöldi blokka í hverju mengi er því blokkir/mengi eða 16/8 = 2,

Breytum 0x12345 í binary þá 00010010001101000101, 20 bitar fyrir 1MB vistfangsrými. Blokkastærðin er 64 bæti sem þýðir að log2(64) = 6 bitar, þar sem við höfum 8 megni þá er mengjavísirinn log2(8) = 3 bitar, þá eru eftirstandandi bitar (20-6-3 = 11) notuð fyrir tagið. Nú getum við skoðað 3 bita mengjavísinn í vistfanginu bitar 6,7,8 frá hægri (teljandi frá 0) það er 101 eða 5 í tugakerfinu og því varpast vistfangið 0x12345 í mengi 5.

**(b)**

**Tölva nokkur hefur 16 bita sýndarvistföng og 12 bita raunvistföng, síðustærðin er 32 bæti. Tölvan hefur 2-vítt TLB með 8 mengjum. (a) Rissið upp skiptingu sýndar- og raunvistfanga í VPN, VPO, PPN, PPO,TLBT og TLBI**

**VPO og PPO:** Þar sem síðustærðin er 32 bæti, þá þurfum við 5 bita til að tákna 32 bæta síðu (25 = 32). Þess vegna eru 5 bitar fyrir VPO og PPO.

**PPN:** 12 bita raunvistfang - 5 bita PPO = 7 bitar fyrir PPN.

**VPN:** 16 bita sýndarvistfang - 5 bita VPO = 11 bitar fyrir VPN.

**TLBI:** Þar sem TLB er 2-vítt með 8 mengjum, þá þurfum við 3 bita til að tákna 8 mengi (2^3 = 8). Þess vegna eru 3 bitar fyrir TLBI.

**TLBT:** Afgangurinn af VPN eftir að TLBI er tekið frá. VPN er 11 bitar - 3 bita TLBI = 8 bitar fyrir TLBT.

**(b) Teiknið upp TLP töflu sem inniheldur gildi 0x24A1. Setjið rétt merki(tag) í löglegt sæti og látið PPN-gildið vera 0x2A og sýnið raunvistfang.**

|  |  |  |
| --- | --- | --- |
| **TLBI** | **TLBT** | **PPN** |
| **5 = (1012)** | **0x24** | **0x2A** |

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**