

Conform Curs 0x00, sistemul de calcul Fugaku are numarul de core-uri

- ☐ a. 5087232
- ☐ b. 537212
- ☐ c. alt raspuns
- ☐ d. 5120000
- ☒ e. 7630848
- ☐ f. 2560000



Răspunsul corect este: 7630848

Cine a spus "The original question, 'Can machines think!' I believe to be too meaningless to deserve discussion"?

- ☐ a. Konrad Zuse
- ☐ b. George Boole
- ☐ c. Blaise Pascal
- ☐ d. Gottfried Wilhelm von Leibniz
- ☐ e. Claude Shannon
- ☐ f. Charles Babbage
- ☐ g. Ada Lovelace
- ☐ h. altcineva
- ☐ i. John von Neumann
- ☒ j. Alan Turing



Răspunsul corect este: Alan Turing

Avem un sistem de calcul cu 2 registrii pe 4 biti fiecare: eax si ebx. In eax avem valoarea 0x7 iar in ebx avem valoarea 0xA. Rezultatul operatiei de adunare $eax \leftarrow eax + ebx$ este in zecimal (numerele sunt naturale):

- ☐ a. 2
- ☐ b. 17
- ☐ c. 5
- ☐ d. 7
- ☐ e. 16
- ☒ f. 1
- ☐ g. 3
- ☐ h. altceva
- ☐ i. 4
- ☐ j. 10



Răspunsul corect este: 1

Avem un sistem de calcul cu 2 registrii pe 8 biti fiecare: eax si ebx. In eax avem valoarea 0x7F iar in ebx avem valoarea 0x1. Rezultatul operatiei de adunare $eax \leftarrow eax + ebx$ este in zecimal (numerele sunt intregi):

- ☐ a. altceva
- ☐ b. -10
- ☒ c. -128
- ☐ d. -57
- ☐ e. -4
- ☐ f. 0
- ☐ g. -16
- ☐ h. -60



Răspunsul corect este: -128

Avem un sistem de calcul cu 2 registrii pe 32 biti fiecare: eax si ebx. In eax avem valoarea 0xFEED iar in ebx avem valoarea 0xBEEF. Rezultatul operatiei $\text{eax} \leftarrow \text{eax AND ebx}$ este in zecimal:

- ☐ a. altceva
- ☐ b. 16636
- ☒ c. 48877
- ☐ d. 16386
- ☐ e. 65261
- ☐ f. 65263
- ☐ g. 32532
- ☐ h. 48879



Răspunsul corect este: 48877

Rezultatul operatiei $(1000 + 1111) \times (1111 + 1111)$ considerand numere naturale este in zecimal:

- ☒ a. altceva
- ☐ b. 1001111011
- ☐ c. 1010110010
- ☐ d. 1111110001
- ☐ e. 1110111001
- ☐ f. 1110111100



Răspunsul corect este: 1010110010

Entropia unor evenimente care au probabilitatile $[1/4, 1/4, 1/4, 1/4]$ este:

- ☒ a. 2
- ☐ b. 2.25
- ☐ c. 0
- ☐ d. 1
- ☐ e. 2.75
- ☐ f. altceva



Răspunsul corect este: 2

Simplificati expresia $(x * y) * (x + z) + (y + !z)$:

- ☒ a. $y + !z$
- ☐ b. $x + y + z$
- ☐ c. $x * y$
- ☐ d. $x + y + !z$
- ☐ e. $(x * y) * (x + z) + (y + !z)$
- ☐ f. altceva



Răspunsul corect este: $y + !z$

Fie numarul FP care are reprezentarea hexa 0x46780000, atunci valoarea zecimala este:

- ☐ a. 1313.3125
- ☐ b. 816.125
- ☐ c. -15872.0
- ☐ d. -816.125
- ☒ e. 15872.0
- ☐ f. altceva
- ☐ g. -1313.3125



Răspunsul corect este: 15872.0

Fie următoarele două instrucțiuni $\%ebx \leftarrow \%edx * \%ecx$, $\%ebx \leftarrow \%eax - \%edx$, ce fel de hazard apare?

- ☒ a. WAW
- ☐ b. RAR
- ☐ c. RAW
- ☐ d. WAR
- ☐ e. altceva



Răspunsul corect este: WAW

Pentru a compara stiintific timpii de rulare pentru doua programe vom folosi:

- ☐ a. media armonica a timpilor de rulare
- ☒ b. mecanismul statistic p-value
- ☐ c. media aritemtica a timpilor de rulare
- ☐ d. nici una
- ☐ e. comparam doar pe complexitatea algoritmilor implementati in cele doua programe
- ☐ f. rulam cele doua programe de cate ori si observam diferentele



Răspunsul corect este: mecanismul statistic p-value

Avem un program care se poate paraleliza in proportie de 90%. De cate procesoare avem nevoie pentru a avea o accelerare a performantei de 6 ori:

- ☒ a. 14
- ☐ b. 11
- ☐ c. 10
- ☐ d. 13
- ☐ e. nu se poate
- ☐ f. 15
- ☐ g. 12



Răspunsul corect este: 14

Ce este adevarat despre setul de instructiuni RISC (in comparatie cu setul de instructiuni CISC)?

- ☒ a. codul sursa scris este mai lung (ca numar de instructiuni) decat cod sursa CISC
- ☐ b. are instructiuni de dimensiune variabila
- ☐ c. altceva
- ☐ d. hardware-ul sistemului de calcul este complicat, software-ul este simplu
- ☐ e. suporta multe metode de adresare



Răspunsul corect este: codul sursa scris este mai lung (ca numar de instructiuni) decat cod sursa CISC

In 2021, Intel a introdus a 12-a generatie de procesoare din gama lor. Procesorul i9-12900K are numarul de core-uri egal cu

- ☐ a. 2
- ☐ b. 8
- ☒ c. 16
- ☐ d. altceva
- ☐ e. 4
- ☐ f. 12



Răspunsul corect este: 16

15 ÎNTREBARE

Corect

Marcat 1,00 din 1,00

🚩 Întrebare cu flag

Intr-o variabila x avem o valoare numerica pozitiva. Cum puteti verifica ca acest numar este par (AND, OR, XOR mai jos sunt operatii pe biti)?

- ☐ a. altceva
- ☐ b. $\text{not}(x \text{ OR } 1)$
- ☐ c. $x \text{ OR } 1$
- ☒ d. $\text{not}(x \text{ AND } 1)$
- ☐ e. $\text{not}(x \text{ XOR } 1)$
- ☐ f. $\text{not } x$
- ☐ g. $x \text{ AND } 1$
- ☐ h. $x \text{ XOR } 1$



Răspunsul corect este: $\text{not}(x \text{ AND } 1)$

16 ÎNTREBARE

Corect

Marcat 1,00 din 1,00

🚩 Întrebare cu flag

In registrul eax avem un numar scris in formatul IEEE FP 32 de biti. Ce face secventa $\text{eax AND } 0x007FFFFFFF$? (AND este operatia pe biti)

- ☐ a. extrage semnul numarului
- ☒ b. extrage mantisa (partea fractionara a numarului)
- ☐ c. extrage exponentul
- ☐ d. verifica daca numarul este zero
- ☐ e. altceva



Răspunsul corect este: extrage mantisa (partea fractionara a numarului)

17 ÎNTREBARE

Corect

Marcat 1,00 din 1,00

🚩 Întrebare cu flag

Avem un sistem de calcul care are urmatoarele flag-uri: Sign Flag (SF, testeaza daca rezultatul este negativ), Zero Flag (ZF, testeaza daca rezultatul este zero), Overflow Flag (OF, testeaza daca rezultatul unei operatii cu semn este overflow) si Unsigned Overflow - denumit si Carry - Flag (CF, testeaza daca rezultatul unei operatii fara semn este overflow). Consideram instructiunea "jle eticheta" (signed less or equal - adica mai mic sau egal cu semn), cum putem scriere aceasta conditie de salt folosind flag-urile de mai sus?

- ☐ a. $\text{not}(\text{SF XOR OF})$
- ☐ b. $\text{not}(\text{SF XOR OF}) \text{ AND } \text{not}(\text{ZF})$
- ☐ c. $\text{not}(\text{CF}) \text{ AND } \text{not}(\text{ZF})$
- ☐ d. altceva
- ☐ e. CF OR ZF
- ☐ f. SF XOR OF
- ☒ g. $(\text{SF XOR OF}) \text{ OR ZF}$



Răspunsul corect este: $(\text{SF XOR OF}) \text{ OR ZF}$

18 ÎNTREBARE

Corect

Marcat 1,00 din 1,00

🚩 Întrebare cu flag

In arhitectura x86, cum se numeste instructiunea care numara cati biti de "1" sunt in reprezentarea binara a unui numar care se afla in registrul eax?

- ☒ a. popcnt
- ☐ b. mov
- ☐ c. lea
- ☐ d. altceva
- ☐ e. lzcnt



Alegeti una dintre cele 8 mari idei din arhitectura calculatoarelor conform Patterson & Hennessy si explicati-o pe scurt.

Performance via Parallelism - se refera la modul prin care executam instructiunile in paralel pentru a avea o performanta mai buna si la faptul ca arhitecturile sunt concepute pentru a oferi mai mult paralelism.

Ce face urmatoarea secventa de cod, compilata cu godbolt x86-64 clang 13, flag -O3?

```
func2(char*):
```

```
mov eax, -1
```

```
.LBB0_1:
```

```
add eax, 1
```

```
cmp byte ptr [rdi], 0
```

```
lea rdi, [rdi + 1]
```

```
jne .LBB0_1
```

```
ret
```

- ☐ a. calculeaza produsul elementelor dintr-un vector
- ☐ b. verifica daca un numar dat este prim
- ☐ c. calculeaza maximul
- ☒ d. calculeaza lungimea unui sir de caractere
- ☐ e. altceva
- ☐ f. calculeaza minimul
- ☐ g. calculeaza suma elementelor dintr-un vector



Răspunsul corect este: calculeaza lungimea unui sir de caractere

Fie programul urmatoar:

.data

<x>:

00 00 00 00

<formatPrintf>:

00 0a 64 25

.text

<f>:

push ebp

mov ebp,esp

push 0x565561e6

push 0x565561f2

mov eax,0x565561c2

jmp eax

mov eax,DWORD PTR [ebp+0x8]

inc eax

mov DWORD PTR [ebp+0x8],eax

pop eax

pop eax

pop ebp

```
mov  eax,0x565561c2
```

```
*****
```

```
jmp  eax
```

```
mov  eax,DWORD PTR [ebp+0x8]
```

```
inc  eax
```

```
mov  DWORD PTR [ebp+0x8],eax
```

```
pop  eax
```

```
pop  eax
```

```
pop  ebp
```

```
ret
```

```
<main>:
```

```
push 0x565561fe
```

```
call 0x565561ad <f>
```

```
pop  edx
```

```
mov  DWORD PTR ds:0x56559008,0x0
```

```
jmp  0x5655620a <final>
```

```
mov  DWORD PTR ds:0x56559008,0x1
```

```
jmp  0x5655620a <final>
```

```
mov  DWORD PTR ds:0x56559008,0x2
```

```
jmp  0x5655620a <final>
```

```
mov  DWORD PTR ds:0x56559008,0x3
```

```
jmp  0x5655620a <final>
```

```
push DWORD PTR ds:0x56559008
```

```
mov  DWORD PTR ds:0x56559008,0x0
```

```
jmp  0x5655620a <final>
```

```
mov  DWORD PTR ds:0x56559008,0x1
```

```
jmp  0x5655620a <final>
```

```
mov  DWORD PTR ds:0x56559008,0x2
```

```
jmp  0x5655620a <final>
```

```
mov  DWORD PTR ds:0x56559008,0x3
```

```
jmp  0x5655620a <final>
```

```
push DWORD PTR ds:0x56559008
```

```
push 0x5655900c
```

```
call 0xf7e1ede0 <printf>
```

```
pop  eax
```

```
pop  eax
```

```
mov  eax,0x1
```

```
xor  ebx,ebx
```

```
int  0x80
```

Linia cu * va fi inlocuita cu: add eax,0xc. Ce se va afisa pe ecran?

- ☐ a. 0
- ☒ b. Segmentation Fault
- ☐ c. 2
- ☐ d. 3
- ☐ e. 1

