Organisation des ordinateurs
Énoncés et solutions de l'examen de première session 2018

Énoncés

[2/20] 1. (a) En français, la probabilité qu'une lettre prise au hasard dans un texte soit un "O", un "C", un "T" ou un "E" est respectivement égale, approximativement, à 5,02%, 3,18%, 5,92% et 12,20%. Par souci de simplicité, on considère que ces probabilités ne dépendent ni de la place des lettres dans les mots, ni de la nature des lettres voisines.

Sous ces hypothèses, on demande de calculer la quantité d'information contenue dans le mot "OCTET".

[1/20] (b) Dans les ordinateurs, pourquoi représente-t-on l'information à l'aide de signaux discrets plutôt que continus ?

[4/20] 2. (a) Quels sont les plus petits et les plus grands nombres représentables à l'aide des encodages

i. entier non signé sur 16 bits ?

ii. entier par complément à deux sur 16 bits ?

iii. en virgule fixe par complément à deux, avec 8 bits avant et 8 bits après la virgule ?

iv. par le procédé IEEE 754 en double précision ?

[1/20] (b) Calculer le produit $-1 \times (-1)$ à l'aide de la représentation par complément à deux des entiers sur 4 bits.

[1/20] (c) Quel est le nombre représenté par la suite de bits

$$\boxed{11000000000000000000000000000000}$$

(il y a 2 bits égaux à 1 suivis de 30 bits égaux à 0), par le procédé IEEE 754 ?

[1/20] 3. (a) A quoi sert la mémoire de masse d'un ordinateur ? Quelles sont les différences entre ce type de mémoire et la mémoire vive ?

[2/20] (b) Expliquer le principe et les modalités d'utilisation de l'adressage indirect indexé de l'architecture x86-64.

[2/20] (c) Le plus simplement possible, décrivez l'effet des instructions x86-64 suivantes :

i. `AND word ptr[R8], 0xff`

ii. `CMP EAX, EAX`

iii. `PUSH qword ptr[RAX]`

iv. `RET`

4. On souhaite programmer une fonction `prefixe_commun(t1, t2, n)` chargée de déterminer la longueur du préfixe commun entre les deux tableaux d'octets pointés par `t1` et `t2`, tous deux de taille `n` $\in [0, 2^{32} - 1]$. En d'autres termes, cette fonction doit déterminer la plus grande valeur k telle que `t1`$[0] =$ `t2`$[0]$, `t1`$[1] =$ `t2`$[1]$, ..., `t1`$[k-1] =$ `t2`$[k-1]$, et retourner cette valeur. Dans le cas où `t1`$[0] \neq$ `t2`$[0]$, la fonction doit retourner 0.

[2/20] (a) Écrire, en pseudocode ou en langage C (au choix), un algorithme permettant de résoudre ce problème.

[4/20] (b) Traduire cet algorithme en assembleur x86-64, en veillant à respecter la convention d'appel de fonctions des systèmes *Unix*.

Exemples de solutions

1. (a) La quantité d'information des lettres "O", "C", "T" et "E" vaut respectivement

$$\log_2 \frac{1}{0{,}0502} \approx 4{,}316 \text{ bits},$$
$$\log_2 \frac{1}{0{,}0318} \approx 4{,}975 \text{ bits},$$
$$\log_2 \frac{1}{0{,}0592} \approx 4{,}078 \text{ bits},$$
$$\log_2 \frac{1}{0{,}1220} \approx 3{,}035 \text{ bits}.$$

Au total, en tenant compte de la présence de deux lettres "T", on obtient donc

$$20{,}483 \text{ bits}.$$

(b) Les signaux discrets permettent de s'affranchir de l'influence du bruit, c'est-à-dire de garantir qu'une valeur transmise sera toujours correctement décodée à sa réception.

2. (a) i. Plus petit nombre : 0. Plus grand nombre : $2^{16} - 1 = 65535$.

ii. Plus petit nombre : $-2^{15} = -32768$. Plus grand nombre : $2^{15} - 1 = 32767$.

iii. Plus petit nombre :

$$-\frac{2^{15}}{2^8} = -2^7 = -128.$$

Plus grand nombre :

$$\frac{2^{15} - 1}{2^8} \approx 127{,}996.$$

iv. La valeur absolue du plus petit et du plus grand nombre possède un exposant égal à 1023 et une mantisse (normalisée) égale à $2 - 2^{52}$. Elle vaut donc

$$2^{1023}\,(2 - 2^{-52}) \approx 2^{1024} \approx 1{,}798\,10^{308}.$$

Le plus petit nombre est donc égal (approximativement) à $-1{,}798\,10^{308}$, et le plus grand à $1{,}798\,10^{308}$.

(b)

$$\begin{array}{cccccc} & 1 & 1 & 1 & 1 \\ \times & 1 & 1 & 1 & 1 \\ \hline & & \boxed{1} & & & \\ & 1 & 1 & 1 & 1 \\ & 1 & 1 & 1 & \\ & 1 & 1 & & & \\ + & 1 & & & & \\ \hline & 0 & 0 & 0 & 1 \end{array}$$

(c) — Il s'agit d'une représentation sur 32 bits, donc en simple précision.

— Le bit de signe est égal à 1 ; le nombre est donc négatif.

— L'exposant e est tel que 10000000 est la représentation binaire non signée de $e + 127$, et est donc égal à 1.

— La mantisse est normalisée et est égale à 1.

— Le nombre représenté vaut donc $-2^1 \times 1 = -2$.

3. (a) La mémoire de masse permet de retenir de façon persistante de grandes quantités de données. Par rapport à la mémoire vive, elle est plus lente, moins chère, et garde son contenu lorsque l'ordinateur est mis hors tension.

(b) Une opérande de la forme

<*taille*> `ptr [` <*base*> `+` <*facteur*> `*` <*index*> `+` <*déplacement*> `]`,

où

— <*taille*> est `qword` (64 bits), `dword` (32 bits), `word` (16 bits) ou `byte` (8 bits),

— <*base*> et <*index*> sont des registres de 64 bits,

— <*facteur*> vaut 1, 2, 4 ou 8.

— <*déplacement*> est une constante,

représente un accès à la mémoire pour lire ou écrire une donnée de la taille spécifiée, à une adresse obtenue en calculant la valeur de l'expression entre les crochets.

(c) i. Cette instruction écrit 0 dans l'octet qui suit celui pointé par le registre `R8`.

ii. Cette instruction lève le drapeau `ZF`, et abaisse les drapeaux `CF`, `SF` et `OF`.

iii. Cette instruction empile une valeur de 64 bits lue en mémoire à l'adresse donnée par le registre `RAX`.

iv. Cette instruction dépile une valeur de 64 bits et effectue un saut à cette adresse.

4. (a)
```
unsigned prefixe_commun(char *t1, char *t2, unsigned n)
{
  unsigned i;

  for (i = 0; i < n; i++)
    if (t1[i] != t2[i])
      break;

  return i;
}
```

(b)
```
                .text
prefixe_commmun: PUSH  RBP
                 MOV   RBP, RSP
                 MOV   RAX, 0
boucle:          CMP   EAX, EDX
                 JAE   fin
                 MOV   CL, byte ptr[RDI + RAX]
                 CMP   CL, byte ptr[RSI + RAX]
                 JNE   fin
                 INC   EAX
                 JMP   boucle
fin:             POP   RBP
                 RET
```