Examen du cours de Fonctionnement des Ordinateurs
1ère Session, Juin 2019
Partie 1

NOM : .... PRENOM : .... SECTION : ....

Consignes à lire impérativement !

L'examen est composé de **2 parties**. Chaque partie dure **2 heures**. Il vous est demandé de respecter les consignes suivantes.

- Commencez par écrire vos **nom**, **prénom** et **section** (math, info, ...) sur chaque feuille, y compris les feuilles de brouillon.
- Laissez vos calculatrice, téléphone portable et notes de cours dans votre sac. Leur usage n'est **pas autorisé**. Pensez à éteindre votre téléphone portable !
- Faites attention à la clarté et à l'organisation de vos réponses. Respectez les règles grammaticales et orthographiques.
- Utilisez pour vos réponses les **cadres** prévus à cet effet. Si davantage d'espace est nécessaire, utilisez le dos de la feuille ou une feuille supplémentaire et indiquez clairement où se situe le restant de la réponse.
- Vous devez terminer cette partie de l'examen avant de pouvoir sortir de la salle (pour aller à la toilette par exemple).
- Toutes les feuilles (énoncé et brouillon) doivent être remises en fin d'examen.
- Vérifiez que vous avez répondu à toutes les questions (il y a **3 questions** dans cette partie).

Question 1 – IEEE 754 (/5)

Cette question concerne la manipulation de nombres flottants représentés en utilisant le standard IEEE 754. La question utilise les mêmes principes que ceux d'IEEE 754 mais, pour des raisons de simplicité, les tailles de mantisse et d'exposant sont beaucoup plus petites. La taille M de la mantisse est 4 bits alors que la taille E de l'exposant est 3 bits. De la même façon qu'avec IEEE 754, un biais B de valeur $2^{E-1} - 1$ est utilisé pour l'exposant. Les arrondis sont effectués selon *round-to-nearest-even*.

Les notations utilisées dans cette question sont les suivantes. Soit x un nombre réel, $\widehat{x}$ dénote une approximation représentable de x. Pour la représentation en virgule flottante d'un nombre, s, e et m désignent respectivement le bit de signe, l'exposant et la mantisse. Finalement, la représentation binaire en virgule flottante est composée de la concaténation de s, e et m, comme illustré ci-dessous.

s	e	m
`1`	`100`	`0101`

Par défaut, une représentation normalisée est utilisée. Cependant, certaines combinaisons de l'exposant et/ou de la mantisse permettent de représenter des valeurs spéciales ou d'utiliser la représentation dénormalisée. Ces combinaisons sont détaillées dans le tableau ci-dessous.

Valeur de e	**Valeur de m**	**Description**
`111`	`1111`	Nombre non représentable
`000`	—	Représentation dénormalisée

Examen du cours de Fonctionnement des Ordinateurs
1ère Session, Juin 2019
Partie 1

NOM : .... PRENOM : .... SECTION : ....

Ce qui vous est demandé :

Q1a Calculez le plus petit nombre positif représentable de façon normalisée.

Q1b Calculez le plus grand nombre positif représentable de façon dénormalisée.

Q1c Donnez une borne maximum sur l'erreur relative (en représentation normalisée).

Q1d Soit le nombre $x = 4,27$. Déterminez la représentation binaire de la plus proche approximation de x, notée $\widehat{x}$.

Q1e Soit le nombre $y = 0,375$. Déterminez la représentation binaire de la plus proche approximation y, notée $\widehat{y}$.

Q1f En partant des représentations de $\widehat{x}$ et $\widehat{y}$ obtenues aux points Q1d et Q1e, calculez $(x - y)$ en effectuant la soustraction $(\widehat{x} - \widehat{y})$ en binaire. Donnez la représentation binaire du résultat, la valeur représentée (notée $\widehat{x-y}$) en décimal et l'erreur absolue Δ_{x-y}.

Pour chacune des questions posées, donnez le justificatif de votre calcul.

Q1a **(plus petit nombre normalisé positif représentable)**

....

....

....

Q1b **(plus grand nombre dénormalisé positif représentable)**

....

....

....

Q1c **(plus grande erreur relative)**

....

....

....

Q1d **(représentation de $\widehat{x}$)**

....

....

....

....

Examen du cours de Fonctionnement des Ordinateurs
1ère Session, Juin 2019
Partie 1

NOM : .... PRENOM : .... SECTION : ....

Q1e **(représentation de $\widehat{y}$)**

....

....

....

....

Q1f **(calcul de $x - y$, valeur de $\widehat{x - y}$ et erreur absolue Δ_{x-y})**

Calcul de $x - y$: ....

....

....

....

....

....

....

Valeur représentée, $\widehat{x - y}$, en décimal : ....

....

....

Erreur absolue Δ_{x-y} : ....

....

....

Question 2 – Mémoire virtuelle (/3)

Dans cette question, il s'agit de jouer le rôle du processeur dans la traduction d'adresses virtuelles en adresses physiques. Le système est doté d'une mémoire physique de 16 KB. L'espace de mémoire virtuelle est fixé à 64 KB. Les pages de mémoire ont une taille de 1 KB. Une TLB (*table lookaside buffer*) de 4 entrées permet d'améliorer les performances d'accès. On utilise le terme terme VPN (*virtual page number*) pour désigner un numéro de page virtuelle et PPN (*physical page number*) pour un numéro de page physique .

Les 10 premières entrées de la table des pages sont montrées à la Table 1. Toutes les autres entrées de la table des pages (non montrées) ont leur bit de validité à 0. Le contenu complet de la TLB est donné en Table 2.

Examen du cours de Fonctionnement des Ordinateurs

1ère Session, Juin 2019

Partie 1

NOM : .... PRENOM : .... SECTION : ....

	PPN	Bit validité
0 :	0	1
1 :	9	1
2 :	5	0
3 :	15	1
4 :	14	0
5 :	11	1
6 :	1	0
7 :	2	0
8 :	5	1
9 :	3	0
⋮	⋮	⋮

TABLE 1 – Contenu de la table des pages.

VPN	PPN	Bit validité
0	0	1
5	8	0
8	5	1
15	7	0

TABLE 2 – Contenu du TLB.

Ce qui vous est demandé :

Q2a Donnez les paramètres suivants de la traduction : la taille (en bits) d'un VPN, d'un PPN et de l'offset à l'intérieur d'une page. Donnez également la taille (en octets) occupée en mémoire par la table des pages.

Le processeur effectue 3 lectures consécutives en mémoire virtuelle aux adresses suivantes : `0x3C91`, `0x23A1` et `0x179E`. Pour chaque accès mémoire, on voudrait déterminer l'adresse physique où sera réalisé l'accès. De plus, pour vous aider à répondre aux questions Q2b-d, il vous est aussi demandé de fournir le VPN et l'offset correspondant à l'adresse virtuelle et d'indiquer si l'accès au TLB se traduit par un « hit » ou « miss » et si une *page fault* est générée.

Q2b Donnez le résultat de l'accès à l'adresse `0x3C91`.

Q2c Donnez le résultat de l'accès à l'adresse `0x23A1`

Q2d Donnez le résultat de l'accès à l'adresse `0x179E`

Q2a **(Paramètres de la traduction)**

Taille VPN (en bits)	
Taille PPN (en bits)	
Taille offset (en bits)	
Taille table des pages (en octets)	

Examen du cours de Fonctionnement des Ordinateurs
1ère Session, Juin 2019
Partie 1

NOM : .... PRENOM : .... SECTION : ....

Q2b **(lecture à l'adresse `0x3C91`)**

VPN	
TLB	Hit ? ☐ ou Miss ? ☐
Page fault ?	☐
Adresse physique (hexa)	

Offset	

Q2c **(lecture à l'adresse `0x23A1`)**

VPN	
TLB	Hit ? ☐ ou Miss ? ☐
Page fault ?	☐
Adresse physique (hexa)	

Offset	

Q2d **(lecture à l'adresse `0x179E`)**

VPN	
TLB	Hit ? ☐ ou Miss ? ☐
Page fault ?	☐
Adresse physique (hexa)	

Offset	

Examen du cours de Fonctionnement des Ordinateurs
$1^{\text{ère}}$ Session, Juin 2019
Partie 1

NOM : .... PRENOM : .... SECTION : ....

Question 3 – Offrez une pause à votre processeur (/2)

L'objectif de cette question est de produire une suite d'instructions MIPS attendant le plus précisément possible un délai donné en millisecondes. Ce délai est égal au temps d'exécution par le processeur de ces instructions. Cette suite d'instructions sera exécutée sur un processeur de fréquence $f = 10\text{MHz}$. On considère que le processeur nécessite deux cycles d'horloge pour exécuter une instruction.

Ce qui vous est demandé :

Q3 Donnez la séquence d'instructions permettant d'attendre un délai prescrit exprimé en millisecondes. La suite d'instructions fournie ne peut utiliser ni timer ni interruption. Le délai à attendre sera placé auparavant dans le registre `a0`. Veillez à écrire votre séquence d'instructions le plus clairement et brièvement possible. Un code simple sera mieux noté qu'un code (inutilement) compliqué.

Q3 **(séquence d'instructions permettant d'attendre un délai prescrit)**

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Examen du cours de Fonctionnement des Ordinateurs
1ère Session, Juin 2019
Partie 2

NOM : .... PRENOM : .... SECTION : ....

Consignes à lire impérativement !

L'examen est composé de **2 parties**. Chaque partie dure **2 heures**. Il vous est demandé de respecter les consignes suivantes.

- Commencez par écrire vos **nom**, **prénom** et **section** (math, info, ...) sur chaque feuille, y compris les feuilles de brouillon.
- Laissez vos calculatrice, téléphone portable et notes de cours dans votre sac. Leur usage n'est **pas autorisé**. Pensez à éteindre votre téléphone portable !
- Faites attention à la clarté et à l'organisation de vos réponses. Respectez les règles grammaticales et orthographiques.
- Utilisez pour vos réponses les **cadres** prévus à cet effet. Si davantage d'espace est nécessaire, utilisez le dos de la feuille ou une feuille supplémentaire et indiquez clairement où se situe le restant de la réponse.
- Vous devez terminer cette partie de l'examen avant de pouvoir sortir de la salle (pour aller à la toilette par exemple).
- Toutes les feuilles (énoncé et brouillon) doivent être remises en fin d'examen.
- Vérifiez que vous avez répondu à toutes les questions (il y a **2 questions** dans cette partie).

Question 1 – Calcul d'une relation de récurrence en MIPS (/6)

L'objectif de cette question est d'implementer en MIPS, sous forme d'une fonction récursive, la relation de récurrence donnée par l'Equation 1. Dans cette équation, a désigne un tableau de n entiers. L'équation permet de déterminer la valeur de s_i pour tout i compris entre 0 et $n-1$. L'opérateur $\div$ désigne la division entière.

$$s_i = \begin{cases} a[0] & \text{si } i = 0 \\ a[1] & \text{si } i = 1 \\ (2a[i] + s_{i-1} + s_{i-2}) \div 4 & \text{sinon} \end{cases} \tag{1}$$

Afin d'illustrer l'Equation 1, la Table 3 donne le contenu d'un tableau a de longueur 6 et les valeurs prises par $a[i]$ et s_i pour $0 \leq i < 6$. Dans l'exemple, on peut notamment observer que $s_2 = (2a[2] + s_1 + s_0) \div 4 = (18+4+13) \div 4 = 8$.

i	0	1	2	3	4	5
$a[i]$	13	4	9	-7	3	-2
s_i	13	4	8	-1	3	-1

TABLE 3 – Exemple de calcul de s_i sur un tableau d'exemple.

Afin de calculer s_i, une fonction nommée `calc_s` doit être implémentée en langage d'assemblage MIPS. Cette fonction a la signature suivante : la valeur de i est passée dans le registre `a0` tandis que l'adresse de base du tableau est passée dans `a1`. Le tableau est composé d'entiers réprésentés en complément à 2 sur 32 bits (4 octets). La valeur de retour (s_i) est retournée dans `v0`. Un exemple d'invocation de `calc_s` pour le tableau $[13, 4, 9, -7, 3, -2]$ et pour $i = 3$ est illustré dans le listing ci-dessous. Cet exemple place les arguments dans `a0` et `a1`, effectue l'appel de `calc_s`, récupère la valeur de s_i dans `v0` et l'affiche à la console à l'aide de l'appel système `syscall`. L'exemple est volontairement incomplet, les parties manquantes étant remplacées par (...).

```
        .data
array:  .word 13, 4, 9, -7, 3, -2, ...

        .text
main:
        (...)
        li      $a0, 3
        la      $a1, array
        jal     calc_s
        move    $a0, $v0
        li      $v0, 1
        syscall
        (...)
        jr      $ra
```

Ce qui vous est demandé :

Q1a L'implémentation de `calc_s` nécessite la lecture des cellules du tableau `array`. Chaque cellule a une taille de 4 octets (32 bits). Supposez que i se trouve dans le registre `t0` et l'adresse de base du tableau `array` dans le registre `a1` et écrivez le code MIPS nécessaire à charger le contenu de `array[`i`]` dans le registre `t1`.

Q1b Fournissez l'implémentation de `calc_s` conformément à la spécififcation donnée ci-dessus.

Note : pour rappel, les multiplications et divisions par 2^n peuvent être réalisées avec les instructions de décalage à gauche et à droite respectivement. Attention cependant, la fonction `calc_s` traite des nombres entiers (donc potentiellement négatifs) !

Q1a **Accès à la cellule i du tableau `array`**

....

....

....

....

....

....

Examen du cours de Fonctionnement des Ordinateurs
1ère Session, Juin 2019
Partie 2

NOM : .... PRENOM : .... SECTION : ....

Q1b **Implémentation de `calc_s`**

NOM : .... PRENOM : .... SECTION : ....

Q1b **Implémentation de `calc_s` (suite)**

Question 2 – Comparateur de nombres entiers (/4)

Cette question vise à implémenter un circuit logique permettant de comparer entre eux deux nombres entiers a et b. Le comparateur dispose à cet effet de $2N$ entrées booléennes correspondant aux deux nombres représentés chacun sur N bits en complément à deux. Ces entrées sont nommées $a_{N-1} \dots a_0$ pour les bits du nombre a et $b_{N-1} \dots b_0$ pour les bits du nombre b. Le comparateur dispose également de deux sorties booléennes définies comme suit :

- eq indique si les deux nombres sont égaux, i.e. $eq \Leftrightarrow (a = b)$
- $less$ indique si a est strictement inférieur à b, i.e. $less \Leftrightarrow (a < b)$

Afin de limiter le nombre d'entrées et la complexité du circuit logique du comparateur, la question considère des nombres représentés sur $N = 2$ bits. Il dispose donc de 4 entrées a_1, a_0, b_1 et b_0.

Examen du cours de Fonctionnement des Ordinateurs

1ère Session, Juin 2019

Partie 2

NOM : .... PRENOM : .... SECTION : ....

Ce qui vous est demandé :

Q2a Etablir la table de vérité de $less$ et eq en fonction des entrées a_1, a_0, b_1 et b_0. La même table est utilisée pour les 2 sorties ($less$ et eq). Dans la colonne intitulée « valeur », indiquez la valeur des nombres représentés en binaire.

Q2b Dérivez une expression logique pour $less$ et eq. Il vous est conseillé d'utiliser pour cela les sommes ou produits canoniques. A vous de choisir entre somme et produit, de façon à minimiser le nombre de termes de l'expression.

Q2c Fournissez le schéma du circuit logique correspondant au test d'égalité. Ce circuit logique possèdera 4 entrées a_1, a_0, b_1 et b_0 ainsi qu'une sortie unique eq. Veillez à utiliser les symboles standards pour exprimer les portes logiques `AND`, `OR` et `NOT`. Veillez également à la propreté et la clarté de votre schéma.

Q2a **(table de vérité)**

Nombre a			**Nombre b**			$less$	eq
valeur	a_1	a_0	**valeur**	b_1	b_0	($a < b$)	($a = b$)

NOM : .... PRENOM : .... SECTION : ....

Q2b **(expression logique pour $less$ et eq)**

....

....

....

....

....

....

....

....

....

....

Q2c **(circuit logique pour eq)**

A Solutions

A.1 Partie 1

A.1.1 Q1 – IEEE 754

Pour rappel, les paramètres de la représentation sont $M = 5$, $E = 3$ et $B = 2^{E-1} - 1 = 3$. Les valeurs d'exposant e `000` et `111` sont réservées respectivement pour la représentation dénormalisée et les nombres non représentables.

a). Plus petite valeur de $\left(1 + \frac{m}{2^M}\right) \cdot 2^{e-B}$ satisfaisant $0 < e < 7$ et $0 <= m < 2^M$. Prendre $m = 0$ et $e = 1$, ce qui donne $1 \cdot 2^{1-B} = 2^{-2} = 0.25$. La représentation correspondante est $s = 0, e = \texttt{001}, m = \texttt{(1)0000}$.

b). Plus grande valeur de $\left(0 + \frac{m}{2^M}\right) \cdot 2^{1-B}$ satisfaisant $0 <= m < 2^M$. Prendre $\frac{2^M-1}{2^M} \cdot 2^{1-B} = \frac{15}{16} \cdot 2^-2 = \frac{15}{64} = 0,234375$. La représentation correspondante est $s = 0, e = \texttt{000}, m = \texttt{(0)1111}$.

c). L'*epsilon machine* (ϵ_M) est une borne supérieure sur l'erreur relative ϵ_x. Avec les paramètres de la représentation, on trouve $\epsilon_M = 2^{-(M+1)} = 2^{-5} = \frac{1}{32} = 0,03125$.

d). Le nombre à représenter est $x = 4.27$. Sa représentation binaire nécessite une infinité de bits `100.0100010100011`... Une approximation est donc utilisée. Comme $M = 4$ et en raison du bit caché, seuls 5 bits peuvent être conservés. La plus proche approximation représentable est $\widehat{x} = 4,25$ (`100.01`).

Comme $\widehat{x}$ est plus grand que le plus petit nombre normalisé représentable, la représentation normalisée est utilisée. Il faut trouver m et e tels que $\widehat{x} = (1 + \frac{m}{2^M}) \cdot 2^{e-B}$ tout en satisfaisant les contraintes de la représentation. Après normalisation, on trouve $\widehat{x} = \left(1 + \frac{1}{16}\right) \cdot 2^2$. Par conséquent, $e = 2 + B = 5$. La représentation correspondante est $s = 0, e = \texttt{101}, m = \texttt{(1)0001}$.

e). Le nombre à représenter est $y = 0,375$. Il est représentable exactement (sans approximation), par conséquent $\widehat{y} = y = 0,375$ (`0.011`). Après normalisation, on trouve $\widehat{y} = \left(1 + \frac{8}{16}\right) \cdot 2^{-2}$ et $e = -2 + B = 1$. La représentation correspondante est $s = 0, e = \texttt{001}, m = \texttt{(1)1000}$.

f). On veut calculer le résultat de $\widehat{x} - \widehat{y}$. A cette fin, on aligne d'abord les exposants de $\widehat{x}$ et $\widehat{y}$. On procède à la soustraction. Le résultat est ensuite normalisé, puis arrondi si nécessaire (ce qui n'est pas le cas ici). Le résultat est $s = 0, e = \texttt{100}, m = \texttt{(1)1111}$.

	e	m	(valeur)
$\widehat{x}$	`101`	`(1)0001`	(4.25)
$\widehat{y}$	`101`	`(0)00011`	(0.375)
$\widehat{x} - \widehat{y}$	`101`	`(0)11111`	(3,875)
$\widehat{x - y}$	`100`	`(1)1111`	(3,875)

La valeur représentée est $\widehat{x - y} = 3,875$. Elle est égale à $\widehat{x} - \widehat{y}$ car aucune erreur d'arrondi n'est introduite durant la soustraction.

L'erreur absolue est égale à $\Delta_{x-y} = \left|(x - y) - (\widehat{x - y})\right| = |3,895 - 3,875| = 0,02$. Cette erreur est uniquement due à l'approximation initiale de x.

A.1.2 Q2 – Mémoire virtuelle

a). Les paramètres du système de mémoire virtuelle sont résumés dans la table suivante. Une adresse virtuelle est composée de 16 bits étant donné qu'il y a 64KB à adresser. Une adresse physique est composée de 14 bits étant donné que la mémoire physique est composée de 16KB. Une page de mémoire a une taille de 1KB, ce qui implique qu'un offset est composé de 10 bits. Finalement, la table des pages fait la correspondance entre un VPN et un PPN. Le VPN est l'index de la table. Il y a donc autant d'entrées dans la table que de VPN différente, c'est à dire $2^6 = 64$ entrées. Une entrée contient un PPN et un bit de validité et occupe donc 6 bits, stockés dans un octet.

Taille du VPN	6 bits
Taille du PPN	4 bits
Taille de l'offset	10 bits
Taille de la table des pages	64 octets

b). Accès à l'adresse `0x3C91`. Le VPN et l'offset valent respectivement `0x0F` et `0x091`. L'entrée du TLB contenant un VPN égal à 15 est invalide. Il s'agit d'un TLB *miss*. L'entrée 15 de la tables des pages est également invalide. L'accès se traduit donc par un *page fault*. Aucun PPN n'est encore associé à ce VPN.

c). Accès à l'adresse `0x23A1`. Le VPN et l'offset valent respectivement `0x08` et `0x3A1`. L'entrée du TLB contenant un VPN égal à 8 est valide. Il s'agit d'un TLB *hit*. Le PPN trouvé dans le TLB vaut `0x5`. L'adresse physique est par conséquent égale à `0x17A1`.

d). Accès à l'adresse `0x179E`. Le VPN et l'offset valent respectivement `0x05` et `0x39E`. L'entrée correspondant du TLB est invalide. Il s'agit d'un TLB *miss*. En revanche, la table des pages contient une entrée 5 valide. Le PPN vaut 11 et l'adresse physique qui en découle est `0x2F9E`.

A.1.3 Q3 – Offrez une pause à votre processeur

Question similaire posée lors de l'examen de Juin 2016.

A.2 Partie 2

A.2.1 Q1 – Calcul d'une relation de récurrence en MIPS

a). Charger la cellule i du tableau nécessite d'effectuer une lecture d'un mot de 32 bits à partir de la mémoire, à l'aide de l'instruction *load word* (`LW`). Il faut auparavant calculer l'adresse mémoire de la cellule. Cette adresse résulte de l'addition de l'adresse de base du tableau et de i fois la taille d'une cellule, c'est-à-dire 4 dans notre cas.

```
SLL  $t0, $t0, 2
ADDU $t0, $t0, $a1
LW   $t1, 0($t0)
```

b). Pour cette question, il est nécessaire d'implémenter une fonction. Celle-ci doit être identifiée par un label `calc_s` et prend seulement 2 arguments : la valeur de i (dans `a0`) et l'adresse de base du tableau (dans `a1`). La fonction doit supporter les deux cas de base (lorsque i vaut 0 ou 1) et le cas récursif lorsque $1 < i < n$. Le cas récursif nécessite deux appels à `calc_s` à l'aide lde l'instruction `JAL` ainsi que la sauvegarde de l'adresse de retour (registre `ra`) et d'autres informations utiles sur la pile. La gestion de la pile nécessite le déplacement du pointeur de pile (registre `sp`) et l'accès à la pile (instructions `LW` et `SW`).

Plusieurs approches sont possibles, mais dans l'implémentation proposée ci-dessous, seul le résultat du 1er appel récursif (s_{i-1}) est stocké sur la pile en plus de `ra`. Le registre `a0` contenant i est toujours restauré à sa valeur d'origine, ce qui évite de devoir le stocker.

```
calc_s:
        bne     $zero, $a0, calc_rec_BC2
        lw      $v0, 0($a1)
        jr      $ra
calc_s_BC2:
        li      $v0, 1
        bne     $v0, $a0, calc_s_REC
        lw      $v0, 4($a1)
        jr      $ra
calc_s_REC:
        addiu   $sp, $sp, -8
        sw      $ra, 0($sp)

        addi    $a0, $a0, -1
        jal     calc_s

        sw      $v0, 4($sp)
        addi    $a0, $a0, -1
        jal     calc_s

        lw      $v1, 4($sp)
        add     $v0, $v0, $v1

        addi    $a0, $a0, 2

        sll     $t0, $a0, 2
        add     $t0, $t0, $a1
        lw      $t1, 0($t0)
        sll     $t1, $t1, 1
        add     $v0, $v0, $t1
        sra     $v0, $v0, 2

        lw      $ra, 0($sp)
        addiu   $sp, $sp, 8
        jr      $ra
```

A.2.2 Q2 – Comparateur de nombres entiers

Question identique posée lors de l'examen de Juin 2016.