

NOM : P I R E S
PRENOM: J U L I E N
INE: 22108030



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|----|
| r2 | ✓ |
| r3 | € |
| r4 | 1 |
| r5 | or |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | ↖ | | |
| 0x004 | | | ↙ | |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

cela.

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

m

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|---|----------|--------|-----|-----|-----|-----|
| _tab: | | _tab | | X | | | |
| _root: | | _root | | | X | | |
| _current: | O | _current | | | | X | |
| | | | | | | | X |
| | | | | | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

| | |
|---------------|--------------------|
| current=NULL; | <i>(Signature)</i> |
|---------------|--------------------|

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

| | |
|----------------------|--------------------|
| root.next = current; | <i>(Signature)</i> |
|----------------------|--------------------|

NOM : T H O M A S
PRENOM: M A R T I N
INE: 1 8 0 0 8 6 0 1



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|---|
| r2 | X |
| r3 | α |
| r4 | β |
| r5 | γ |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | α | | |
| 0x004 | | | β | |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

où

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4,0x1(zero)** ? Expliquez pourquoi.

où

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

oui

non

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|---|----------|--------|-------|-----|-----|-----|
| _tab: | X | _tab | | | X | | |
| _root: | X | _root | | | | X | |
| _current: | | _current | | | | X | |
| | | | | 0x000 | | X | |
| | | | | 0x004 | | | X |
| | | | | 0x008 | | | X |
| | | | | 0x00C | | | |
| | | | | 0x010 | | | |
| | | | | 0x014 | | | X |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

oui

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

non

NOM : A M I A R D
PRENOM: A N T H O N Y
INE:



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|---|
| r2 | ✓ |
| r3 | ✗ |
| r4 | 0 |
| r5 | 1 |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | 1 | | |
| 0x004 | | | 0 | 0 |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

Cm

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

ou

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

car

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|----------|----------|--------|-------|-----|-----|-----|
| _tab: | <i>N</i> | _tab | 0 | | X | | |
| _root: | <i>D</i> | _root | 0 | | | 1 | |
| _current: | <i>N</i> | _current | 1 | | | | 0 |
| | | | | 0x000 | | | |
| | | | | 0x004 | | | |
| | | | | 0x008 | | | |
| | | | | 0x00C | | | |
| | | | | 0x010 | | | |
| | | | | 0x014 | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

M

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

M

NOM : L E D O U R N I E R
PRENOM: G U I L C A U M E
INE: 1 8 0 0 0 7 3 6



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|---|
| r2 | 0 |
| r3 | 1 |
| r4 | 0 |
| r5 | 1 |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | X | | X | |
| 0x004 | | X | | |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

Non

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

On

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

corr

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|--|----------|--------|-----|-----|-----|-----|
| _tab: | | _tab | | X | | | |
| _root: | | _root | X | | | | |
| _current: | | _current | | | | X | |
| | | | | | | | X |
| | | | | | | X | |
| | | | | | X | | |
| | | | | | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

corr

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

corr

NOM : L A L A N D E - M A R C H A N D
PRENOM: A R T H U R
INE: 1 6 0 0 8 5 4 8



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD
addi r3,zero,0x8FFF
stb r2,0x6(zero)
ldh r4,0x2(zero)
srai r5,r3,8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|-----|
| r2 | oui |
| r3 | nn |
| r4 | x |
| r5 | |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | x | | |
| 0x004 | | | x | |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

oui ou n

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

ja p~

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

Cela

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|--|----------|--------|-------|-----|-----|-----|
| _tab: | | _tab | X | | | | |
| _root: | | _root | | | | | |
| _current: | | _current | | | | | |
| | | | | 0x000 | | | |
| | | | | 0x004 | | | |
| | | | | 0x008 | | | |
| | | | | 0x00C | | X | |
| | | | | 0x010 | | | X |
| | | | | 0x014 | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

~

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

✓

NOM : GOUTIERREZ ANORÉ

PRENOM: ALBAN

INE: 17011707



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2,0x12ABCD  
addi r3,zero,0x8FFF  
stb r2,0x6(zero)  
ldh r4,0x2(zero)  
srai r5,r3,8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|---|
| r2 | ○ |
| r3 | 1 |
| r4 | ✗ |
| r5 | |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | ✗ | | |
| 0x004 | ✗ | | | |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

oui

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4,0x1(zero)** ? Expliquez pourquoi.

non

non

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

peut être

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|--|----------|--------|-------|-----|-----|-----|
| _tab: | | _tab | X | | | | |
| _root: | | _root | | | X | | |
| _current: | | _current | | | | | |
| | | | | 0x000 | | | |
| | | | | 0x004 | | X | |
| | | | | 0x008 | | | |
| | | | | 0x00C | | | |
| | | | | 0x010 | | | |
| | | | | 0x014 | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

Oui

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

N

NOM : S O U V I N
PRENOM: T O M
INE: 2 0 1 0 9 8 1 8



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|---|
| r2 | X |
| r3 | 0 |
| r4 | 1 |
| r5 | X |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | X | | | |
| 0x004 | | | | X |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

où

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

Am

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

120

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-------|--|----------|--------|-----|-----|-----|-----|
| _tab: | | _tab | | | | | |
| | | _root | | | X | | |
| | | _current | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

et re

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

rr

NOM : D E L A D N A Y
PRENOM: S U L I E R
INE: 2 0 2 0 0 4 2 6



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|-----|
| r2 | buu |
| r3 | |
| r4 | |
| r5 | |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | x | | | |
| 0x004 | | | | x |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

non

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

fa

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

12

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|--|----------|--------|-----|-----|-----|-----|
| _tab: | | _tab | | 2 | | | |
| _root: | | _root | | | | | |
| _current: | | _current | | | 2 | | 2 |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |
| | | | | | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

00000000

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

00000000

NOM : G I R A U D E T
PRENOM: THÉO
INE: 18001342



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|---|
| r2 | X |
| r3 | |
| r4 | 0 |
| r5 | 1 |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | X | | |
| 0x004 | | | | X |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

oui

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

erreur

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

tab

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| | .data | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|-------|----------|--------|-------|-----|-----|-----|
| _tab: | | _tab | | | X | | |
| _root: | | _root | | | | X | |
| _current: | | _current | | | | | |
| | | | | 0x000 | | | |
| | | | | 0x004 | | | |
| | | | | 0x008 | | | |
| | | | | 0x00C | | | |
| | | | | 0x010 | | | |
| | | | | 0x014 | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

current=NULL;

f

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

root.next = current;

w

NOM : Q U T D E L C E U R
PRENOM: B E N S A M I N
INE: 2 0 1 1 1 S 3 3



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|----|
| r2 | x |
| r3 | on |
| r4 | ~ |
| r5 | ok |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | x | | |
| 0x004 | | | x | |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

en

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

en

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

700

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|--|----------|--------|-----|-----|-----|-----|
| _tab: | | _tab | | | | | |
| _root: | | _root | | | | | |
| _current: | | _current | | | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

```
current=NULL;
```

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

```
root.next = current;
```

NOM : H U L O T
PRENOM: V A L E N T I N
INE: 2 0 2 0 0 8 0 4



NOM DU MODULE

Février 2022

- Durée de l'épreuve : 1h00 heures
- Polycopiés de cours autorisés
- Calculatrices non connectées autorisées.
- Le barème est donné à titre purement indicatif.

Les réponses doivent être portées sur ce document, sous peine d'être ignorées par le correcteur.

Exercice 1 : Jeu d'instructions Nios II (5 points)

On s'intéresse à un processeur NIOS-II qui exécute les instructions d'un programme représenté ci-dessous. L'état de la mémoire au début de l'exécution du programme est représenté dans le tableau ci-après.

```
movia r2, 0x12ABCD
addi r3, zero, 0x8FFF
stb r2, 0x6(zero)
ldh r4, 0x2(zero)
srai r5, r3, 8
```

| | 00 | 01 | 10 | 11 |
|-------|------|------|------|------|
| 0x000 | 0xF1 | 0x23 | 0xC3 | 0x84 |
| 0x004 | 0x40 | 0x53 | 0x9D | 0x62 |

Q1 : Complétez les tables ci-dessous en précisant les valeurs des registres ainsi que l'état de la mémoire à l'issue de l'exécution de ce programme.

| | |
|----|---|
| r2 | 1 |
| r3 | 0 |
| r4 | 1 |
| r5 | 0 |

| | 00 | 01 | 10 | 11 |
|-------|----|----|----|----|
| 0x000 | | X | | |
| 0x004 | | | X | |

Q2 : Pouvait-on utiliser l'instruction **movi** à la place de **movia** dans le programme ci-dessus ? Si non, expliquez pourquoi.

C'est pas possible car il n'y a pas d'adresse dans l'opcode.

Q3 : Que se passe-t-il lors de l'exécution de l'instruction **ldw r4, 0x1(zero)** ? Expliquez pourquoi.

Le contenu de la mémoire à l'adresse 0x001 est lu et placé dans le registre r4.

Exercice 2 : programmation assembleur (10 points)

On s'intéresse ici à la traduction du code C ci-dessous vers le langage machine NIOS II

```
struct node {  
    int* data;  
    struct node* next;  
};  
  
int tab[] = {1,45,234};  
struct node root = { .data=tab, .next=NULL};  
struct node* current = &root;
```

Q1 : Donnez la taille (en octets) de l'espace mémoire occupé par chacune des variables globales **tab**, **root** et **current** (vous justifierez votre réponse).

bloc

Q2 : Complétez les directives d'assemblage NIOS-II permettant d'allouer et d'initialiser les variables globales **tab**, **root** et **current**. Déduisez en les valeurs associées aux identificateurs **_tab**, **_root**, **_current** ainsi que le contenu de la mémoire au début du programme.

| .data | | Ident | valeur | 0x0 | 0x1 | 0x2 | 0x3 |
|-----------|--|----------|--------|-------|-----|-----|-----|
| _tab: | | _tab | | | | | |
| _root: | | _root | | | X | | |
| _current: | | _current | | | | X | |
| | | | | 0x000 | | | |
| | | | | 0x004 | | X | |
| | | | | 0x008 | | | X |
| | | | | 0x00C | | | |
| | | | | 0x010 | | | |
| | | | | 0x014 | | | |

Q3 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous.

| | |
|---------------|----------|
| current=NULL; | <i>w</i> |
|---------------|----------|

Q4 : Traduisez (en langage d'assemblage NIOS-II) l'instruction ci-dessous (on fera l'hypothèse que la valeur associée à **_current** est codée sur 16 bits)

| | |
|----------------------|----------|
| root.next = current; | <i>w</i> |
|----------------------|----------|