

INF2170

Organisation des ordinateurs et assembleur

Examen Final

Dimanche 29 avril 2018

Hiver 2018

Durée : 2 h 30

Aucun document n'est autorisé ni aucune feuille de brouillon.

L'usage de la calculatrice ou d'appareils électroniques est interdit.

Détachez, répondez et rendez seulement la dernière page (feuille des réponses). Conservez l'examen.

Chacune des 20 questions n'a qu'une seule bonne réponse.

1. Questions de cours

Question 1 : L'adresse d'un tableau est passée comme paramètre au sous-programme en utilisant la pile. Quel est le mode d'adressage à utiliser dans un sous-programme pour accéder en lecture ou en écriture à un élément du tableau stocké dans la mémoire principale ?

A d
B i
C s

D sx
E x
F n

G sf
H sxf

Question 2 : Choisissez une instruction où le pointeur de pile n'est pas impliqué durant son exécution.

A MOVFLGA
B SUBSP
C CALL

D MOVSPA
E RET7
F ADDSP

G DECI
H RETTR
I RET0

Question 3 : Sur combien d'octets la suite d'instructions ci-dessous est codée en langage machine?

DECI n,d
LDA n,d
NEGA
NOP3
ASLA
STOP
n: .WORD 0

A 8 octets
B 6 octets
C 7 octets

D 11 octets
E 9 octets
F 10 octets

G 12 octets
H 16 octets
I 18 octets

2. Représentation en virgule flottante

Question 4 : Convertir le nombre $5_{10} * 2^{-129}$ selon la norme IEEE-754, simple précision et donner le premier octet de cette représentation.

Rappel : 1 bit de signe, 8 bits d'exposant (pôle 127), 23 bits de mantisse.

- A 00XXXXXX₁₆
- B 42XXXXXX₁₆
- C 40XXXXXX₁₆

- D 22XXXXXX₁₆
- E 80XXXXXX₁₆
- F 85XXXXXX₁₆

- G 82XXXXXX₁₆
- H 48XXXXXX₁₆
- I 41XXXXXX₁₆

Question 5 : Convertir le nombre $0.101_2 * 2^{-32}$ selon la norme IEEE-754, simple précision et donner le premier octet de cette représentation.

- A 27XXXXXX₁₆
- B 2FXXXXXX₁₆
- C 1FXXXXXX₁₆

- D 7EXXXXXX₁₆
- E 72XXXXXX₁₆
- F 37XXXXXX₁₆

- G 3FXXXXXX₁₆
- H AFXXXXXX₁₆
- I 00XXXXXX₁₆

Question 6 : Soit C2100000₁₆ la représentation hexadécimal d'un nombre flottant simple précision selon la norme IEEE-754. Déterminez dans quelle fourchette se situe le nombre.

- A < -100
- B >= -100 et < -60
- C >= -60 et < -45

- D >= -45 et < -20
- E >= -20 et < 5
- F >= 5 et < 15.5

- G >= 15.5 et < 17.5
- H >= 17.5 et < 100
- I >= 100

Question 7 : Parmi les représentations suivantes trouver celle de $-\infty$ selon la norme IEEE-754.

- A 7F800000₁₆
- B FF800000₁₆
- C 3F800000₁₆

- D 00000001₁₆
- E 80000001₁₆
- F FF800001₁₆

- G 80000000₁₆
- H 00000000₁₆
- I 7F800001₁₆

Feuille brouillon

127

	128	64	32	16	8	4	2	1
	0	1	1	1	1	1	1	1

A : 10
B : 11
C : 12
D : 13
E : 14
F : 15

5 0101 1.2 -128

$$\begin{array}{r} 64 \\ + 32 \\ \hline 96 \end{array}$$

$$\begin{array}{r} 16 \\ + 16 \\ \hline 32 \end{array}$$

120

0011 1114

3

-32
-33

7F
00

33 HOK = 0010 0001

2 1

7F
-21
5E

C 2 1 0

1	1	0	0	0	0	0	0	0
1	1	0	0	1	0	1	0	1
8	4	2						

716
84
-7F
5

1.00100

10 0100

2 4

3. Sous-programmes

Qu'affiche le programme suivant ?

```
CALL x
CALL z
STOP
x: CHARO 'X',i
CHARO ',',i
CALL y
RETO
y: CHARO 'Y',i
CHARO ',',i
RETO
z: CHARO 'Z',i
CHARO ',',i
RETO
.END
```

Call x
call y
call z

x, y, z,

Question 8 :

- A X,Z,
- B X,Y,Z
- C X,Y,

- D X,Y,Z,
- E X,Z
- F X,Y

- G X,Z,Y
- H X,Z,Z,
- I X,Z,Y,

Qu'affiche le programme suivant ?

```
CALL a
CALL b
CALL c
STOP
a: DECO 12,i
RETO
b: DECO -34,i
BR c
c: CHARO '**',i
RETO
.END
```

call a
call b BRC
call c

12 -34 ** *

Question 9 :

- A 12-34*
- B 12*-34
- C 12-*34

- D 12*-34**
- E 12**-34
- F 12*-34*

- G Aucune de ces réponses
- H 12-34****...
- I 12-34**

Soit le programme suivant :

```

SUBSP 16,i
LDX 0,i
CPX 10,i
BRGE loop1
DECI tab,sx
BRLT done
ADDX 2,i
BR loop
loop:
LDX 10,i
STX 10,s
MOVSPA
ADDA tab,i
STA 12,s
CALL funct
DECO 14,s
ADDSP 16,i
STOP
tab:
.EQUATE 0
;
funct:
SUBSP 6,i
STA savA,s
STX savX,s
LDX 0,i
STX ct,s
f_loop:
CPX sz,s
BRGE f_fin
LDA a,sxf
BREQ f_fin
LDA 1,i
ANDA a,sxf
BREQ cmpt
f_1:
ADDX 2,i
BR f_loop
cmpt:
LDA ct,s
ADDA 1,i
STA ct,s
BR f_1
f_fin:
LDA ct,s
STA s1,s
LDA savA,s
LDX savX,s
RET6
s1:
.EQUATE 22
a:
.EQUATE 20
sz:
.EQUATE 18
savA:
.EQUATE 0
savX:
.EQUATE 2
ct:
.EQUATE 4
.END

```

$X = 0 \text{ Z } 10 \quad A = SP + Tab_i$

$X = 0 \text{ Z } 00$

-1000

1000

-15

45

0

123

-1000

34

1000

0

-15

12

45

15

6

10

16

SP+Tab*i*

0

14

SP+Tab*i*

10

Question 10 : Qu'affiche ce programme lorsque l'on saisit les valeurs :

-1000

1000

-15

45

0

- | | |
|---|------|
| A | -875 |
| B | 1050 |
| C | 35 |

- | | |
|---|--------------------|
| D | Rien n'est affiché |
| E | 5 |
| F | 2065 |

- | | |
|---|------|
| G | -340 |
| H | 0 |
| I | 127 |

Question 11 : Qu'affiche ce programme lorsque l'on saisit les valeurs :

123

34

0

12

15

- | | |
|---|----|
| A | 0 |
| B | 3 |
| C | 15 |

- | | |
|---|-----|
| D | 123 |
| E | 1 |
| F | 12 |

- | | |
|---|--------------------|
| G | 2 |
| H | 34 |
| I | Rien n'est affiché |

Question 12 : Qu'affiche ce programme lorsque l'on saisit les valeurs :

1

2

3

4

5

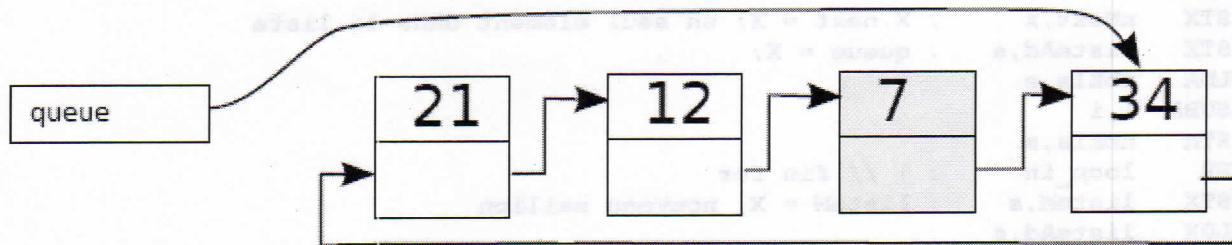
- | | |
|---|-----|
| A | 135 |
| B | 2 |
| C | 24 |

- | | |
|---|-------|
| D | 12345 |
| E | 5 |
| F | 3 |

- | | |
|---|--------------------|
| G | 1 |
| H | Rien n'est affiché |
| I | 4 |

4. Appels de fonctions, paramètres et variables dans la pile

Soit le programme incomplet suivant implémentant la création d'une liste chaînée et de son affichage. La liste est représentée par un pointeur sur le dernier élément et le pointeur `next` de celui-là pointe vers le premier élément. Voir la figure.



Le programme est incomplet. Les parties manquantes sont remplacées par des séquences de lettres de A à D. Une fois complété, ce programme affichera la liste. Des nouveaux maillons sont insérés à la fin de la liste.

```
;-----  
; Créer une liste de n éléments; insertion à la fin de la liste;  
; La liste est représentée par un pointeur sur le dernier élément - queue;  
; queue->next pointe sur le premier élément;  
; Afficher la liste chaînée créée.  
;  
main:    STRO msg1,d  
          DECI tmp,d  
          LDX tmp,d  
;  
; Q13  
; Empilage des paramètres et réservation de l'espace pour une valeur de  
; retour  
AAAAAAAAAAA  
;  
        STRO msg2,d  
        CALL liste      ; paramètre: #nbEls,valeur de retour: #listeAd  
        STRO msg3,d  
        CALL aff        ; paramètre: #listeAd  
        ADDSP 4,i       ; libérer #listeAd #nbEls  
        STOP  
tmp:     .BLOCK 2      ; #2d  
msg1:    .ASCII "Entrez un nombre d'éléments:\n\x00"  
msg2:    .ASCII "Entrez les éléments:\n\x00"  
msg3:    .ASCII "La liste:\n\x00"  
;  
; Prologue du sous-programme LISTE: créer une liste de n éléments  
; IN:      PP+0 = nombre d'éléments  
; OUT:    PP+2 = adresse de la queue  
;  
; Q14  
; Déclaration des symboles  
BBBBBBBBBBB  
;  
;  
liste:   SUBSP 6,i      ; #listeM #listeSA #listeSX  
          STA listeSA,s  
          STX listeSX,s  
          LDX 0,i  
          STX listeAd,s ; queue = null  
          LDA nbEls,s
```

```

loop_in: CPA 0,i
        BRLE out ; for(cpt = nbEls; cpt>0; cpt--) {
; Q15
; Allocation dynamique d'un maillon dans le tas
CCCCCCCCCCCC
        DECI mVal,x ; X.val = getInt();
        LDA listeAd,s
        BRNE nV
        STX mNext,x ; X.next = X; un seul élément dans la liste
        STX listeAd,s ; queue = X;
cn:    LDA nbEls,s
        SUBA 1,i
        STA nbEls,s
        BR loop_in ; } // fin for
nV:    STX listeM,s ; listeM = X, nouveau maillon
        LDX listeAd,s
        LDA mNext,x ; queue->next
        LDX listeM,s
        STA mNext,x ; listeM->next = queue->next
        LDX listeAd,s ; queue->next = listeM
        LDA listeM,s
        STA mNext,x
        STA listeAd,s ; queue = listeM
        BR cn
out:   LDX listeSX,s ; restaurer X
        LDA listeSA,s ; restaurer A
        RET6 ; depiler #listeSX #listeSA #listeM
;-----;
; aff:   Afficher la liste
; IN:    PP+6 = adresse de la queue de la liste
;-----;
queue: .EQUATE 6 ; #2h adresse de la queue
curr:  .EQUATE 0 ; #2h élément courant, variable locale
;
aff:   SUBSP 2,i ; empiler #curr
        LDX queue,s
        LDX mNext,x ; curr = queue->next = tête
        STX curr,s
loop_out: CPX queue,s ; curr == queue ?
        BREQ fin ; for (curr=head; curr!=queue; curr=curr.next) {
        DECO mVal,x
        CHARO ' ',i ; print(curr.val + " ");
        LDX mNext,x
        BR loop_out ; } // fin for
fin:   DECO mVal,x ; afficher le dernier élément
        RET2 ; depiler #curr
***** Structure de liste d'entiers
; Une liste est constituée d'une chaîne de maillons.
; Chaque maillon contient une valeur et l'adresse du maillon suivant
; La fin de la liste est marquée arbitrairement par l'adresse 0
mVal:   .EQUATE 0 ; #2d valeur de l'élément dans le maillon
mNext:  .EQUATE 2 ; #2h maillon suivant (null (aka 0) pour fin de liste)
mLength: .EQUATE 4 ; taille d'un maillon en octets
***** operator new
; Precondition: A contient le nombre d'octets à allouer
; Postcondition: X contient un pointeur aux octets alloués
new:    LDX hpPtr,d ; pointeur retourné
; Q16
; Allocation dynamique
DDDDDDDDDDDDDD
hpPtr:  .ADDRSS heap ; adresse aux octets libres
heap:   .BLOCK 1 ; le premier octet du tas
END

```

Question 13 : Quelle séquence d'instructions faut-il placer à la place de AAAAA... pour faire un empilage des paramètres ?

A	SUBSP 2,i STX 0,s	D	SUBSP 4,x STX 0,s	G	SUBSP 6,i STA 0,s
B	SUBSP 4,i STA 0,s	E	SUBSP 4,x STA 0,s	H	ADDSP 4,i STX 0,s
C	SUBSP 4,i STX 0,s	F	SUBSP 6,i STX 0,s	I	ADDSP 4,i STA 0,s

Question 14 : Quelle séquence d'instructions faut-il mettre à la place de BBBB... comme déclaration des symboles.

listeNb – le paramètre du sous-programme, listeAd – la valeur de retour, listeSA, listeSX et listeM – variables locales.

A	listeAd: .EQUATE 10 nbEls: .EQUATE 8 listeM: .EQUATE 4 listeSA: .EQUATE 2 listeSX: .EQUATE 0	D	listeAd: .EQUATE 8 nbEls: .EQUATE 6 listeM: .EQUATE 4 listeSA: .EQUATE 2 listeSX: .EQUATE 0	G	listeAd: .EQUATE 2 nbEls: .EQUATE 0 listeM: .EQUATE 8 listeSA: .EQUATE 6 listeSX: .EQUATE 4
B	listeAd: .EQUATE 10 nbEls: .EQUATE 8 listeM: .EQUATE 6 listeSA: .EQUATE 4 listeSX: .EQUATE 2	E	listeAd: .EQUATE 8 nbEls: .EQUATE 6 listeM: .EQUATE 0 listeSA: .EQUATE 2 listeSX: .EQUATE 4	H	listeAd: .EQUATE 8 nbEls: .EQUATE 0 listeM: .EQUATE 6 listeSA: .EQUATE 2 listeSX: .EQUATE 4
C	listeAd: .EQUATE 8 nbEls: .EQUATE 6 listeM: .EQUATE 4 listeSA: .EQUATE 2 listeSX: .EQUATE 0	F	listeAd: .EQUATE 2 nbEls: .EQUATE 0 listeM: .EQUATE 10 listeSA: .EQUATE 8 listeSX: .EQUATE 6	I	listeAd: .EQUATE 10 nbEls: .EQUATE 0 listeM: .EQUATE 2 listeSA: .EQUATE 6 listeSX: .EQUATE 4

Question 15 : Quelle séquence d'instructions est nécessaire pour compléter le code qui effectue l'allocation dynamique d'un nouveau maillon de la liste (CCC...) ?

A	LDX mLength,i CALL new	D	LDA mLength,d LDX mVal,i CALL new	G	LDA mLength,d CALL new
B	LDA mLength,i LDX mVal,i CALL new	E	LDA mLength,i LDX mVal,x CALL new	H	LDX mLength,d CALL new
C	LDX mLength,i LDA mVal,i CALL new	F	LDA mLength,i CALL new	I	LDA mLength,s CALL new

Question 16 : Quelle séquence d'instructions doit être placée à DDD... pour compléter le code de l'allocation dynamique sur le tas ?

A	ADDA hpPtr,d STA hpPtr,d RETI	D	ADDA hpPtr,s STA hpPtr,sx RETO	G	ADDA hpPtr,i STA hpPtr,d RETO
B	ADDA hpPtr,x STA hpPtr,s RETO	E	ADDA hpPtr,d STX hpPtr,d RETO	H	ADDX hpPtr,i STA hpPtr,d RETO
C	ADDX hpPtr,d STX hpPtr,d RETO	F	ADDA hpPtr,d STA hpPtr,i RETO	I	ADDA hpPtr,d STA hpPtr,d RETO

Feuille brouillon

5. Modes d'adressage dans la pile

Soit la liste d'assemblage suivante :

Object Addr	code	Symbol	Mnemon	Operand	Comment

0000	02			; Positionner SP à t	
0001	E1001D		MOVSPA		A = SP
0004	69001D		STA	spval,d	spval
0007	60001F		SUBSP	spval,d	V = 81
			ADDSP	t,i	
			;Début		
000A	530001		CHARO	1,s ; Q17	
000D	540002		CHARO	2,sf ; Q18	
0010	C80003		LDX	3,i	
0013	560002		CHARO	2,sx ; Q19	
0016	C80001		LDX	1,i	
0019	570006		CHARO	6,sxf ; Q20	
001C	00		STOP		
001D	0000	spval:	.BLOCK	2	
		;Haut de la pile			
001F	61	t:	.BYTE	'a'	
0020	62	y:	.BYTE	'b'	
0021	0024	u:	.WORD	0x0024	
0023	63	v:	.BYTE	'c'	
0024	64	w:	.BYTE	'd'	
0025	001F	z:	.WORD	0x001F	
0027			.END		



Symbol table

Symbol	Value	Symbol	Value
spval	001D	t	001F
u	0021	v	0023
w	0024	y	0020
z	0025		

Question 17 : Quel est le premier caractère affiché (CHARO 1,s)

A	35 ₁₀	D	a	G	0x0020
B	b	E	23 ₁₆	H	#
C	c	F	d	I	0

Question 18 : Quel est le second caractère affiché (CHARO 2,sf)

A	35 ₁₀	D	a	G	0x0020
B	b	E	23 ₁₆	H	#
C	c	F	d	I	0

Question 19: Quel est le troisième caractère affiché (CHARO_{2,sx})

- | | |
|---|------------------|
| A | 35 ₁₀ |
| B | b |
| C | c |

- | | |
|---|------------------|
| D | a |
| E | 23 ₁₆ |
| F | d |

- | | |
|---|--------|
| G | 0x0020 |
| H | # |
| I | 0 |

Question 20: Quel est le quatrième caractère affiché (CHARO_{6,sxf})

- | | |
|---|------------------|
| A | 35 ₁₀ |
| B | b |
| C | c |

- | | |
|---|------------------|
| D | a |
| E | 23 ₁₆ |
| F | d |

- | | |
|---|--------|
| G | 0x0020 |
| H | # |
| I | 0 |

Question 17: Quel est le premier caractère affiché (charo_{2,sx})

- | | |
|---|--------|
| G | 0x0050 |
| H | # |
| I | 0 |

- | | |
|---|------------------|
| D | 5 |
| E | 23 ₁₆ |
| F | q |

- | | |
|---|------------------|
| A | 35 ₁₀ |
| B | p |
| C | c |

Question 18: Quel est le second caractère affiché (charo_{2,sx})

- | | |
|---|--------|
| G | 0x0050 |
| H | # |
| I | 0 |

- | | |
|---|------------------|
| D | 5 |
| E | 23 ₁₆ |
| F | q |

- | | |
|---|------------------|
| A | 35 ₁₀ |
| B | p |
| C | c |

6. Annexes

6.1 39 instructions Pep/8

Spécif.	Instruct.	Significations	Modes d'adressage	Codes instr. conditions
00000000	STOP	Arrête l'exécution du programme		
00000001	RETTR	Retour d'interruption	(nibble)	
00000010	MOVSPA	Placer SP dans A		
00000011	MOVFLGA	Placer NZVC dans A		
0000010a	BR	Branchemet inconditionnel	i,x	
0000011a	BRLE	Branchemet si inférieur ou égal	i,x	
0000100a	BRLT	Branchemet si inférieur	i,x	
0000101a	BREQ	Branchemet si égal	i,x	
0000110a	BRNE	Branchemet si non égal	i,x	
0000111a	BRGE	Branchemet si supérieur ou égal	i,x	
0001000a	BRGT	Branchemet si supérieur	i,x	
0001001a	BRV	Branchemet si débordement	i,x	
0001010a	BCR	Branchemet si retenue	i,x	
0001011a	CALL	Appel de sous-programme	i,x	
0001100r	NOTr	NON bit-à-bit du registre		NZ
0001101r	NEGr	Opposé du registre		NZV
0001110r	ASLr	Décalage arithmétique à gauche du registre		NZVC
0001111r	ASRr	Décalage arithmétique à droite du registre		NZC
0010000r	ROLr	Décalage cyclique à gauche du registre C		
0010001r	RORr	Décalage cyclique à droite du registre C		
001001nn	NOPn	Interruption unaire pas d'opération		
00101aaa	NOP	Interruption non unaire pas d'opération	i	
00110aaa	DECI	Interruption d'entrée décimale	d,n,s,sf,x,sx,sxf	NZV
00111aaa	DECO	Interruption de sortie décimale	i,d,n,s,sf,x,sx,sxf	
01000aaa	STRO	Interruption de sortie de chaîne	d,n,sf	
01001aaa	CHARI	Lecture caractère	d,n,s,sf,x,sx,sxf	
01010aaa	CHARO	Sortie caractère	i,d,n,s,sf,x,sx,sxf	
01011nnn	RETn	Retour d'un appel avec n octets locaux		
01100aaa	ADDSP	Addition au pointeur de pile (SP)	i,d,n,s,sf,x,sx,sxf	NZVC
01101aaa	SUBSP	Soustraction au pointeur de pile	i,d,n,s,sf,x,sx,sxf	NZVC
0111raaa	ADDR	Addition au registre	i,d,n,s,sf,x,sx,sxf	NZVC
1000raaa	SUBR	Soustraction au registre	i,d,n,s,sf,x,sx,sxf	NZVC
1001raaa	ANDr	ET bit-à-bit du registre	i,d,n,s,sf,x,sx,sxf	NZ
1010raaa	ORr	OU bit-à-bit du registre	i,d,n,s,sf,x,sx,sxf	NZ
1011raaa	CPr	Comparer registre	i,d,n,s,sf,x,sx,sxf	NZVC
1100raaa	LDr	Placer 1 mot dans registre	i,d,n,s,sf,x,sx,sxf	NZ
1101raaa	LDBYTEr	Placer octet dans registre (0-7)	i,d,n,s,sf,x,sx,sxf	NZ
1110raaa	STr	Ranger registre dans 1 mot	d,n,s,sf,x,sx,sxf	
1111raaa	STBYTER	Ranger registre (0-7) dans 1 octet	d,n,s,sf,x,sx,sxf	

6.2 8 directives Pep/8

Directive	Signification
.BYTE	Réserve 1 octet mémoire avec valeur initiale.
.WORD	Réserve 1 mot mémoire avec valeur initiale.
.BLOCK	Réserve un nombre d'octets mis à zéro.
.ASCII	Réserve l'espace mémoire pour une chaîne de caractères (ex: "Chaîne").
.ADDRSS	Réserve 1 mot mémoire pour un pointeur.
.EQUATE	Attribue une valeur à une étiquette.
.END	Directive obligatoire de fin d'assemblage qui doit être à la fin du code.
.BURN	Le programme se terminera à l'adresse spécifiée par l'opérande.

6.3 Codes ASCII importants (hexadécimaux)

Codes ASCII	Caractères
00	Caractère NUL
0A	Caractère de saut de ligne '\n' (Enter)
20	Espacement ' '
30	Premier chiffre '0'
41	Premier caractère alphabétique majuscule 'A'
61	Premier caractère alphabétique minuscule 'a'

6.4 Adressages

Mode	aaa	a	Lettres	Opérande
Immédiat	000	0	i	Spec
Direct	001		d	mem[Spec]
Indirect	010		n	mem[mem[Spec]]
Sur la pile	011		s	mem[PP+Spec]
Indirect sur la pile	100		sf	mem[mem[PP+Spec]]
Indexé	101	1	x	mem[Spec + X]
Indexé sur la pile	110		sx	mem[PP+Spec+X]
Indirect indexé sur pile	111		sxf	mem[mem[PP+Spec]+X]

6.5 Registres

Symbol	r	Description	Taille
N		Négatif	1 bit
Z		Nul (Zero)	1 bit
V		Débordement (Overflow)	1 bit
C		Retenue (Carry)	1 bit
A	0	Accumulateur	2 octets (un mot)
X	1	Registre d'index	2 octets (un mot)
PP		Pointeur de pile (SP)	2 octets (un mot)
CO		Compteur ordinal (PC)	2 octets (un mot)
		Spécificateur d'instruct	1 octet
Spec		Spécificateur d'opérand	2 octets (un mot)