

# Organisation des ordinateurs

## Examen type

*Livres fermés. Durée : 3 heures 1/2.*

*Veillez répondre à chaque question sur une feuille séparée sur laquelle figurent nom, prénom et section. Soyez bref et concis, mais précis. Les calculatrices non programmables sont autorisées.*

**Le contenu de cet examen type n'est qu'indicatif! En première et en seconde session, les questions d'examen peuvent porter sur toutes les matières vues au cours.**

1. Une bande magnétique destinée au stockage de données comprend 512 pistes parallèles. Sur chacune d'entre elles, les données sont encodées par l'orientation de domaines magnétiques disposés bout à bout. Chaque domaine magnétique occupe une longueur de 100 nm, et possède deux orientations possibles.
  - [2/20] (a) Si la longueur de la bande magnétique est de 600 m, quelle quantité d'information brute contient-elle ? (Détaillez votre raisonnement.)
  - [1/20] (b) Pour la mémoire de masse d'un ordinateur, pourquoi préfère-t-on employer un disque dur plutôt qu'une telle bande magnétique ?
  
- [4/20] 2. (a) Quels sont les nombres représentés par la constante de 16 bits 0xFF80 dans les représentations
  - i. non signée,
  - ii. par complément à 1,
  - iii. par complément à 2,
  - iv. par complément à 2 en virgule fixe, avec 6 bits après la virgule.
- [1/20] (b) Calculer l'addition  $-2 + (-3)$  à l'aide de la représentation par complément à 1 des entiers sur 4 bits.
- [1/20] (c) Dans la représentation IEEE 754, quelle est l'utilité de la représentation dénormalisée des mantisses ?

- [2/20] 3. (a) A quoi sert l'ALU d'un processeur ? Comment cette ALU est-elle connectée aux autres composants de celui-ci ?
- [1/20] (b) En quelques mots, qu'est-ce que la mémoire cache d'un processeur ? De quel type de mémoire est-elle constituée ?
- [2/20] (c) Le plus simplement possible, décrivez l'effet des instructions x86-64 suivantes :
- i. XOR R10, R10
  - ii. MOV word ptr[RAX], BX
  - iii. JNS qword ptr[8\*RDI]
  - iv. PUSH 0x1234
4. On souhaite programmer une fonction `remplir(t, n, v)` chargée de remplir le tableau d'octets pointé par `t`, de taille `n`  $\in [0, 2^{32} - 1]$ , avec la valeur `v`  $\in [0, 255]$ .
- [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*.

# Annexe

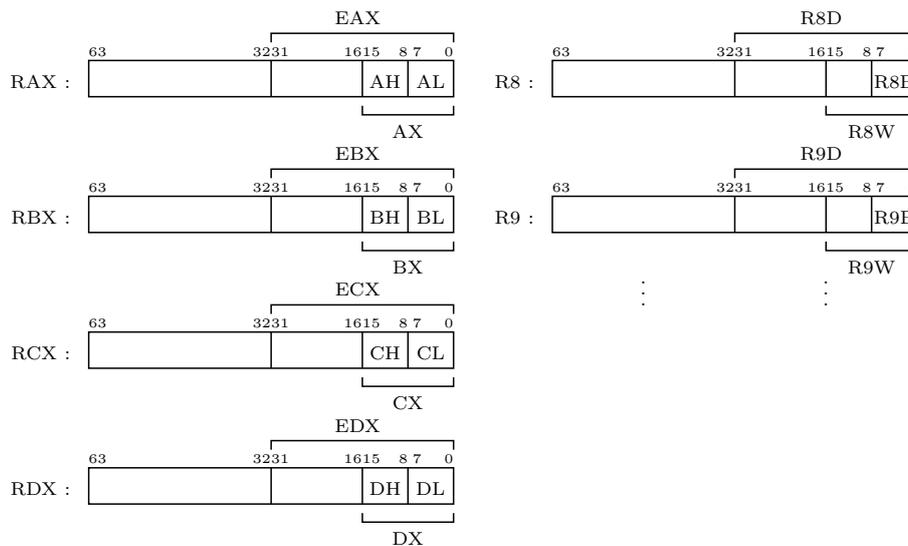
## Code ASCII

20		30	0	40	@	50	P	60	'	70	p
21	!	31	1	41	A	51	Q	61	a	71	q
22	”	32	2	42	B	52	R	62	b	72	r
23	#	33	3	43	C	53	S	63	c	73	s
24	\$	34	4	44	D	54	T	64	d	74	t
25	%	35	5	45	E	55	U	65	e	75	u
26	&	36	6	46	F	56	V	66	f	76	v
27	'	37	7	47	G	57	W	67	g	77	w
28	(	38	8	48	H	58	X	68	h	78	x
29	)	39	9	49	I	59	Y	69	i	79	y
2A	*	3A	:	4A	J	5A	Z	6A	j	7A	z
2B	+	3B	;	4B	K	5B	[	6B	k	7B	{
2C	,	3C	<	4C	L	5C	\	6C	l	7C	
2D	-	3D	=	4D	M	5D	]	6D	m	7D	}
2E	.	3E	>	4E	N	5E	^	6E	n	7E	~
2F	/	3F	?	4F	O	5F	-	6F	o		

## UTF-8

- $[0, 0x7F]$  :  $0b_6b_5 \dots b_0$
- $[0x80, 0x7FF]$  :  $110b_{10}b_9 \dots b_6$   $10b_5b_4 \dots b_0$
- $[0x800, 0xFFFF]$  :  $1110b_{15}b_{14}b_{13}b_{12}$   $10b_{11}b_{10} \dots b_6$   $10b_5b_4 \dots b_0$
- $[0x10000, 0x10FFFF]$  :  $11110b_{20}b_{19}b_{18}$   $10b_{17}b_{16} \dots b_{12}$   $10b_{11}b_{10} \dots b_6$   $10b_5b_4 \dots b_0$

## Registres x86-64



## Modes d'adressage des instructions x86-64

MOV, ADD, SUB, CMP, AND, OR, XOR	
Op. 1	Op. 2
<i>reg</i>	<i>imm</i>
<i>mem</i>	<i>imm</i>
<i>reg</i>	<i>reg</i>
<i>reg</i>	<i>mem</i>
<i>mem</i>	<i>reg</i>

XCHG	
Op. 1	Op. 2
<i>reg</i>	<i>reg</i>
<i>reg</i>	<i>mem</i>
<i>mem</i>	<i>reg</i>

INC, DEC, NOT, POP	
Op. 1	
<i>reg</i>	
<i>mem</i>	

MUL, IMUL, PUSH, JMP, Jxx, LOOP, CALL	
Op. 1	
<i>imm</i>	
<i>reg</i>	
<i>mem</i>	

## Drapeaux affectés par les instructions x86-64

	CF	ZF	SF	OF
MOV, XCHG, NOT, PUSH, POP, JMP, Jxx, LOOP, CALL, RET	—	—	—	—
ADD, SUB, CMP	✓	✓	✓	✓
AND, OR, XOR	0	✓	✓	0
INC, DEC	—	✓	✓	✓
MUL, IMUL	✓	?	?	✓

## Instructions de saut conditionnel x86-64

Instruction	Condition
JC	CF = 1
JNC	CF = 0
JZ	ZF = 1
JNZ	ZF = 0
JS	SF = 1
JNS	SF = 0
JO	OF = 1
JNO	OF = 0

Instruction	Condition
JE	$op1 = op2$
JNE	$op1 \neq op2$
JG	$op1 > op2$ (valeurs signées)
JGE	$op1 \geq op2$ (valeurs signées)
JL	$op1 < op2$ (valeurs signées)
JLE	$op1 \leq op2$ (valeurs signées)
JA	$op1 > op2$ (valeurs non signées)
JAЕ	$op1 \geq op2$ (valeurs non signées)
JB	$op1 < op2$ (valeurs non signées)
JBE	$op1 \leq op2$ (valeurs non signées)

## Convention d'appel de fonctions Unix

- Six premiers arguments : Registres RDI, RSI, RDX, RCX, R8 et R9.
- Valeur de retour : Registre RAX.
- Registres à préserver : RBX, RBP, R12, R13, R14 et R15.