Organisation des ordinateurs
Examen de seconde session 2019

Livres fermés. Durée : 3 heures 1/2.

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

[2/20] 1. (a) Une machine est constituée de deux récipients opaques R_1 et R_2 entre lesquels sont réparties mille billes identiques. Chaque bille a une probabilité égale de se trouver dans R_1 ou dans R_2, indépendamment des autres billes. La machine comprend également une balance de précision située sous R_1, affichant en permanence le nombre de billes que ce récipient contient.

Si la balance affiche 0, quelle quantité d'information fournit-elle sur l'état de la machine ? (Donner la réponse sous forme numérique, et justifier votre calcul.)

[2/20] (b) Une station météorologique est équipée de deux capteurs : un thermomètre affichant la température par pas de 0,5°, et un pluviomètre renseignant la quantité de précipitations en millimètres avec un chiffre après la virgule. On suppose que la température est indépendante des précipitations, que la température affichée suit une distribution uniforme entre $-20°$ et $40°$, et que la valeur renseignée par le pluviomètre suit une distribution uniforme entre 0 et 100 mm.

Quelle quantité de mémoire, exprimée en octets, est-elle nécessaire pour enregistrer le bulletin météorologique de cette station ? (Donner la réponse sous forme numérique, et justifier votre calcul.)

[2/20] (c) Dans les circuits d'un ordinateur, pourquoi préfère-t-on utiliser des signaux discrets plutôt que continus ? Pourquoi choisit-on habituellement des signaux possédant deux valeurs nominales ?

[2/20] 2. (a) Représenter le nombre -16 sur 8 bits par valeur signée et par complément à deux.

[1/20] (b) Représenter le nombre $-1{,}25$ en virgule fixe par complément à deux, avec 4 chiffres avant la virgule et 4 chiffres après la virgule.

[1/20] (c) Représenter le nombre 2^{-128} dans le format IEEE754 en simple précision, en exprimant le résultat en hexadécimal.

[1/20] (d) Exprimer la valeur du nombre entier dont la représentation par complément à deux forme la suite de bits $b_3 b_2 b_1 b_0$.

[1/20] (e) Effectuer l'addition $(-2) + (-5)$ en représentant les nombres par complément à un sur 4 bits.

3. Écrire des fragments de code assembleur x86-64 réalisant les opérations suivantes. (On supposera chaque fois que la pile est initialement correctement configurée.)

[1/20] (a) Permuter les deux valeurs de 64 bits situées au sommet de la pile.

[1/20] (b) Appeler 100 fois une fonction `f`, sans lui fournir d'argument, en employant la convention d'appel des systèmes *Unix*.

[2/20] (c) Déterminer si le contenu du registre AL correspond au premier octet d'un caractère représenté sur 16 bits selon l'encodage UTF-8. À l'issue de cette opération, AH vaudra 1 dans le cas positif, et 0 sinon.

4. On souhaite programmer une fonction `count_ok(str)` comptant le nombre d'occurrences de la chaîne de caractères "ok" dans une chaîne `str`. Cette dernière contient des caractères encodés en ASCII, et est terminée par un zéro.

[1/20] (a) Écrire, en pseudocode ou en langage C (au choix), un algorithme permettant de résoudre ce problème.

[3/20] (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

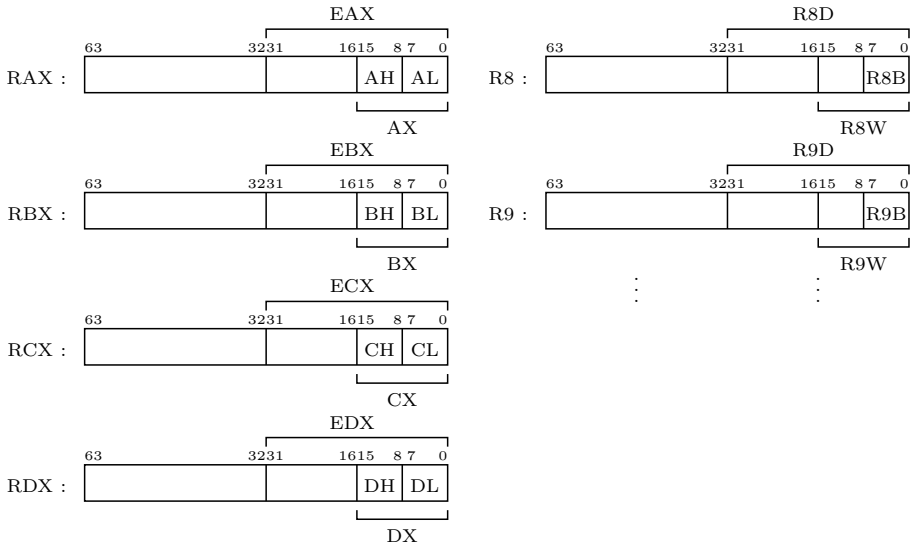
— [0, 0x7F] : $\boxed{0b_6b_5\dots b_0}$

— [0x80, 0x7FF] : $\boxed{110b_{10}b_9\dots b_6}$ $\boxed{10b_5b_4\dots b_0}$

— [0x800, 0xFFFF] : $\boxed{1110b_{15}b_{14}b_{13}b_{12}}$ $\boxed{10b_{11}b_{10}\dots b_6}$ $\boxed{10b_5b_4\dots b_0}$

— [0x10000, 0x10FFFF] : $\boxed{11110b_{20}b_{19}b_{18}}$ $\boxed{10b_{17}b_{16}\dots b_{12}}$ $\boxed{10b_{11}b_{10}\dots b_6}$ $\boxed{10b_5b_4\dots b_0}$

Registres x86-64



Modes d'adressage des instructions x86-64

`MOV, ADD, SUB, CMP, AND, OR, XOR`	
Op. 1	Op. 2
reg	*imm*
mem	*imm*
reg	*reg*
reg	*mem*
mem	*reg*

`XCHG`	
Op. 1	Op. 2
reg	*reg*
reg	*mem*
mem	*reg*

`INC, DEC, NOT, POP`
Op. 1
reg
mem

`MUL, IMUL, PUSH, JMP, Jxx, LOOP, CALL`
Op. 1
imm
reg
mem

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)
`JAE`	$op1 \geq op2$	(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.