IFT 1215 — Examen final — Hiver 2017
Introduction aux systèmes informatiques

Professeur : Michel Boyer

Durée : 3 heures.

Directives :

- *Documentation permise : une feuille manuscrite format* A4 *ou lettre* (8.5 × 11).
- *Aucun appareil électronique autorisé (calculatrice, téléphone, iPod, etc).*
- *Répondre sur le questionnaire dans l'espace disponible. Utiliser le verso si nécessaire.*
- *Attention, les questions ne sont pas nécessairement en ordre de difficulté.*
- *Total :* 100 × 0.4 = 40% *de la note finale.*

Nom _____

Prénom _____

Matricule _____

Question 1	/20
Question 2	/25
Question 3	/15
Question 4	/10
Question 5	/20
Question 6	/10
Total	/100

Signature : _____

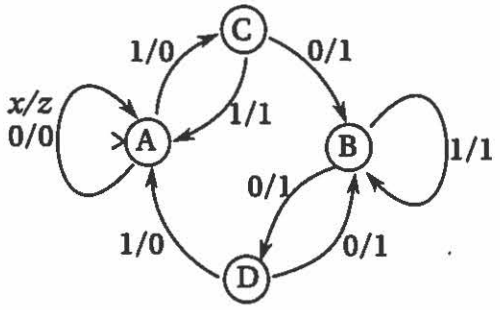
1. Circuits séquentiels (20 points)

a) (6 points) Donnez un ***diagramme d'états (automate)*** qui prend en entrée, séquentiellement, un à un, des bits, et met 1 en sortie lorsque les trois derniers bits lus sont 000, 110, 101 ou 011 et qui met 0 en sortie sinon. Il faut bien sûr mettre 0 en sortie quand moins de deux bits ont été lus.

b) (4 points) Donnez quatre suites d'entrées distinctes (du temps 1 au temps 10) qui donnent avec un automate tel que spécifié en a) les sorties spécifiées ci-dessous.

temps	1	2	3	4	5	6	7	8	9	10
$Entrées_1$										
$Entrées_2$										
$Entrées_3$										
$Entrées_4$										
Sorties	0	0	1	1	1	1	1	1	1	1

c) (10 points) Le diagramme suivant décrit un automate avec les quatre états A, B, C et D, qui prend en entrée un bit x et retourne en sortie le bit z.



Complétez la table d'états (***table de transition***) de cet automate en spécifiant dans chaque case les valeurs s'/z où s' est le nouvel état et z la sortie :

$s \backslash x$	$x=0$	$x=1$
A		
B		
C		
D		

Donnez un encodage binaire des états et écrivez la table de vérité permettant d'implanter l'automate ci dessus avec bascules D selon votre codage (la table doit préciser les valeurs de z et de d_i pour chaque q_i et x où les bits q_i encodent les états de d_i les entrées D des bascules).

État	A	B	C	D
Code binaire				

Table de vérité :

2. La machine LMC (25 points)

La machine LMC a ses instructions et valeurs numériques (en complément à 10) codées avec trois décimales, ses adresses sont de deux décimales, de 00 à 99. Si on désigne par M[xy] le contenu de la cellule mémoire d'adresse xy (xy de 00 à 99), le mnémonique de l'assembleur, le code opération et l'action effectuée par chaque instruction est alors comme suit :

Mnémonique	Code	Action
ADD	$1xy$	M[xy] est additionné au contenu de A
SUB	$2xy$	M[xy] est soustrait du contenu de A
STO	$3xy$	le contenu de A est copié dans M[xy]
LDA	$5xy$	M[xy] est copié dans A
BR	$6xy$	Brancher sans condition à l'adresse xy
BRP	$8xy$	Brancher à l'adresse xy si A ≥ 0
BRZ	$7xy$	Brancher à l'adresse xy si A est égal à 0
IN	901	L'entrée est mise dans A
OUT	902	Le contenu de A est copié en sortie.
HLT	000	L'exécution s'arrête
DAT		Met une valeur numérique à l'adresse courante.

L'exécution utilise les registres A (accumulateur), MAR (memory address register), MDR (memory data register), PC (compteur ordinal) et IR (registre instruction) ; on note IR[adr] la partie adresse (contenant la valeur xy) du registre instruction. Attention, tous les programmes débutent à l'adresse 00.

a) (5 points) Donnez les transferts de registres pour le fetch et l'exécution de l'instruction ADD de la LMC (telle que décrite ci-dessus).

b) (3 points) Traduisez dans le langage machine de la LMC le code assembleur suivant en indiquant bien en première colonne les adresses où sont mises les instructions (les programmes débutent toujours à l'adresse 00) ; les entrées de la colonne "Code LMC" sont chacune trois chiffres décimaux.

```
A     IN
      BRP B
      HLT
B     OUT
      BR A
```

Adresse	Code LMC
00	901

c) (6 pts) Complétez le tableau suivant en écrivant le contenu de IR, PC, MAR, MDR et A immédiatement après que chacune des instructions du programme en b) ait terminé son exécution et ce jusqu'à la fin d'exécution du programme ; écrivez aussi l'entrée lue ou la sortie obtenue si l'instruction cause une entrée ou une sortie. Les entrées sont successivement 2, 5 et -1.

IR	PC	MAR	MDR	A	Entrée	Sortie

d) (4 points) Quelle valeur faut-il mettre en entrée pour que le programme suivant mette 130 en sortie ? Justifiez votre réponse.

```
    IN
    STO A
A   DAT 000
    OUT
    HLT
B   DAT 025
```

e) (7 points) Écrire en assembleur LMC un programme qui lit en entrée des entiers positifs ou nuls et retourne en sortie la valeur maximale lue et s'arrête dès qu'une entrée négative est tapée. Si la première entrée lue est négative, le programme doit retourner -1 et s'arrêter. [Se fait en 15 lignes de code.]

3. Mémoire cache (15 points).

Dans cette question $K = 2^{10}$, $M = 2^{20}$, $G = 2^{30}$. Supposons une mémoire adressable de 4G octets (les adresses ont 32 bits) et une mémoire cache (cache direct) de 1M octets au total, et dont les lignes sont de 256 octets.

a) (4 pts) Combien la mémoire cache contient-elle de lignes et combien de bits ont respectivement l'index et l'étiquette (tag) ?

# bits étiquette	# bits index	# bits offset

Justifiez votre réponse :

Découpez correctement l'adresse 89ABCDEF sous forme étiquette-index-déplacement (en notation hexadécimale)

étiquette	index	déplacement

b) (6 pts) Si l'adresse 89ABCDEF a sa copie en cache, si le « dirty bit » de sa ligne est levé (1) dire en détails tout ce qui est modifié (en mémoire et cache) si le cache est direct et le programme accède maintenant à l'adresse 99ABCDEF

c) (2 pts) Si ce même ordinateur a une mémoire dont le temps de latence, i.e. le temps pour accéder au premier mot, est de 50 ns ; le temps de transfert des mots suivants est de 10 ns, quel est le temps de chargement d'une ligne de cache ?

d) (3 pts) Sous les mêmes hypothèses quel est le temps moyen d'accès au cache si le taux de succès (hit rate) est de 85%, si le temps d'accès au cache est de 4 ns et si lors d'un défaut (miss) de cache, il suffit de d'abord charger la ligne puis ensuite d'y accéder. Écrivez votre réponse sous forme de formule puisque vous n'avez pas la calculatrice.

4. Adressage et format des instructions (10 points)

Sur la WKC, instruction `PUSH n` et `PUSH n(FP)` sont codées comme suit

$1n_{14}n_{13}\ldots n_0$	`PUSH` n
$010n_{12}n_{11}\ldots n_0$	`PUSH` n`(FP)`

avec, dans les deux cas, une valeur de n signée en complément à deux.

a) (3 pts) Donnez les 16 bits qui encodent chacune des instructions suivantes

(i) `PUSH -1`

(ii) `PUSH 0(FP)`

(iii) `PUSH -1(FP)`

b) (3 pts) Quel l'interval des entiers relatifs qui peuvent être miss sur la pile avec un seul appel à l'instruction `PUSH n` ? [Utiliser des puissances de 2 pour exprimer votre réponse].

c) (4 pts) Ecrivez le code assembleur WKC avec nombre en notation hexadécimale pour empiler chacune des valeurs suivantes en une ou deux instructions WKC. Notez que quand on écrit `PUSH 0xABCD`, l'assembleur prend les 15 bits les moins significatifs du nombre passé en paramètre (ici `010 1011 1100 1101`) et empile le nombre de 16 bits obtenu après extension de signe. L'instruction `CHBIT15` permet ensuite de changer le bit de signe si besoin est.

(i) Empiler `0111 1000 1001 1010`

(ii) Empiler `1100 1101 1110 1111`

5. Infixe et postfixe (20 points).

a) Soit l'expression $((x-y)*(x+z))-(((x+y)*(x+z))-((y-z)*(x-z)))$.

(i) (5 points) Donnez sa représentation sous forme arborescente.

(ii) (5 points) En déduire sa représentation en notation postfixe.

b) (i) (3 points) Traduisez en postfixe l'expression $F(z, y) - G(x + y)$

(ii) (7 points) Écrivez le code `.asm` pour la WKC correspondant à la fonction définie par

```
function H(x,y,z){
  return F(z,y) - G(x+y)
}
```

où F et G sont des fonctions supposées déjà définies (toutes les fonctions ont des arguments entiers et retournent une valeur entières ; note : les instructions de la WKC sont en appendice).

6. La machine WKC, implantation et exécution (10 points).

a) Un programme pour la WKC contient les instructions suivantes aux adresses 32 à 37

```
PUSH  2(FP)
PUSH  3(FP)
PROUT
PUSH  F
CALL  2
PROUT
```

ainsi que le code suivant pour F :

```
F: PUSH 1(FP)
   PROUT
   PUSH 2(FP)
   PROUT
   PUSH 3(FP)
   POP  0(FP)
   RETFROM 2
```

Si FP contient 1030 et SP contient 1040 juste avant l'exécution de l'instruction à l'adresse 37 et si les valeurs aux adresses 1030 à 1034 sont 10, 11, 12, 13, 14, quelles seront dans l'ordre les 4 sorties obtenues par l'exécution du code des adresses 32 à 37 ?

b) Écrire les transferts de registres pour l'instruction ADD de la WKC (décrite en page 13). La WKC a pour registres PC, SP, FP, IR, MAR, MDR et un registre Y utile pour les calculs arithmétiques (supposez le fetch déjà complété et faites uniquement les transferts propres à l'instruction ADD).

Instructions de la WKC

ADD	dépile deux données et empile leur somme
MUL	dépile deux données et empile leur produit
DIV	dépile y, depile x, puis empile x/y
SUB	dépile y, depile x, puis empile x-y
AND	dépile deux valeurs et empile leur AND
OR	dépile deux valeurs et empile leur OR
NOT	remplace le dessus de pile par sa négation
NEQ	dépile deux valeurs, empile 1 si elles sont inégales, sinon empile 0
EQ	dépile deux valeurs, empile 1 si elles sont égales, sinon empile 0
LT	dépile y, dépile x, empile 1 si x < y, sinon empile 0
GT	dépile y, dépile x, empile 1 si x > y, empile 0 sinon
BRIF	dépile une adresse, dépile un "booléen", branche à l'adresse si le "booléen" n'est pas 0.
BR	dépile une adresse et y branche, i.e. met dans le registre PC l'adresse dépilée
READIN	lit en entrée et empile la valeur lue
PROUT	dépile une valeur et l'imprime en sortie
NOOP	ne fait rien
PUSH n	empile l'entier n (i.e. incrémente SP et met n à l'adresse dans SP)
PUSH n(FP)	empile la valeur qui est contenue à l'adresse FP+n
POP n(FP)	dépile et met la valeur dépilée dans l'adresse FP+n
CALL n	dépile l'adresse du code où l'on va brancher empile la valeur contenue dans le registre PC (déjà incrémenté) empile la valeur contenue dans le registre FP met dans le registre FP la valeur SP - n - 1 met dans le registre PC l'adresse qui avait été dépilée.
RETFROM n	met dans PC le contenu de l'adresse FP+n met dans SP la valeur contenue dans FP met dans FP la valeur à l'adresse FP+n+1
CHBIT15	inverse le bit le plus significatif (le bit de signe) de la valeur en dessus de pile.
HALT	arrête l'exécution

Sur cette machine 0 représente Faux et toute valeur non nulle donne Vrai.