

Contrôle Continu UE INF401 : Architectures des ordinateurs

Mars 2022, durée 1 h 30

Document : 1 A4 R/V personnel manuscrit autorisé ; caleuses et téléphones portables interdits.
La plupart des questions sont indépendantes, si vous avez du mal avec l'une, passez à la suivante.
Tant que possible, indiquez bien tous les détails et justifiez vos réponses.
Le barème est donné à titre indicatif.

1 Numération et opération en binaire et en complément à 2 (6 points)

Rappel important : en machine, les entiers relatifs (\mathbb{Z}) sont représentés selon la méthode dite du complément à 2 (C2), il ne faut pas confondre cette méthode de représentation avec l'opération dite de complémentation.

- Déterminer le bon nombre de bits pour représenter toutes les valeurs suivantes puis donner la représentation binaire de ces entiers relatifs avec le nombre de bits choisi (codage en complément à 2) : $(+222)_{10}$, $(-201)_{10}$. **(2 points)**
- Donner la représentation binaire et la valeur décimale des 2 entiers relatifs suivants codés en complément à 2 sous forme hexadécimale : $(021E)_{16}$ et $(FAFA)_{16}$. **(2 points)**
- Effectuer, en donnant tous les détails (dont les retenues), les deux opérations suivantes, sur 1 octet : $(0011\ 1010)_2 + (0110\ 1110)_2$ et $(1110\ 1011)_2 - (0101\ 1011)_2$.
Pour chacune d'elle, donner une interprétation du résultat pour la représentation binaire usuelle et une interprétation pour le codage en complément à 2.
Pour chacune d'elles, donner la valeur des indicateurs (Z, N, C et V).
Pour la soustraction, effectuer l'opération par addition du complémentaire. **(2 points)**

2 Rotations multiples d'un texte (8 points)

Cet exercice a pour objectif de programmer n rotations d'un texte. Exemple pour 3 rotations : "Bonjour le monde" => "onjour le mondeB", "njour le mondeBo", "jour le mondeBon". L'algorithme suivant est donc à traduire (le texte est dans la variable txt).

```
1. EcrChaine(txt)
2. ALaLigne
3. si txt[0]==0 alors
4.   EcrChaine(''Chaine vide'')
5.   ALaLigne
6. sinon si n==0 alors
7.   EcrChaine(''Pas de rotation'')
8.   ALaLigne
9. sinon
```

```

10.   pour j= 1 .. n faire
11.     c := txt[0]
12.     i := 1
13.     tant que txt[i]!=0 faire
14.       t[i-1] := t[i]
15.       i := i+1
16.     fin tant que
17.     t[i-1] := c
18.     EcrNdecimal32(j)
19.     EcrChaine(' '>')
20.     EcrChaine(txt)
21.     ALaLigne
22.   fin pour
23.   finsi

```

Dans l'algorithme, les variables `n`, `i` et `j` sont des entiers naturels sur 1 mot, les fonctions `EcrChaine` et `ALaLigne` correspondent aux fonctions du fichier `es.s` étudié en cours et en TP. Le fonctionnement des fonctions d'`es.s` est rappelé en annexe du sujet.

Vous allez maintenant traduire l'algorithme en assembleur `ARM` en vous basant sur le squelette de code donné ci-dessous et l'allocation globale des registres suivante : `i` dans `R5`, `j` dans `R6`, `c` dans `R7`.

```

.data
txt: .asciz "Bonjour le monde"
    .balign 4
n: .word 3

.rodata
msgVide: .asciz "Chaine vide"
msg0: .asciz "Pas de rotation"
msgDonne: .asciz "=>"

.text
.global main
main:
    push {lr}

        @ partie manquante

    pop {lr}
    bx lr

LD_txt: .word txt
LD_n: .word n
LD_msgVide: .word msgVide
LD_msg0: .word msg0
LD_msgDonne: .word msgDonne

```

Questions :

- (d) Quelle est la place mémoire occupée par le texte ? (0,5 point).
- (e) Rappelez le sens de la directive `.balign`. Selon le cas, expliquer pourquoi la directive `.balign` est nécessaire ou au contraire pourquoi on pourrait l'omettre dans le cas présent. (1 point).
- (f) Donner le code ARM pour les affichages (lignes 18, 19, 20, 21) (1 point).
- (g) Donner le code ARM complet pour l'exécution du test de la ligne 3 ; si le test est vrai, le branchement sera vers l'étiquette `affichageTexteVide` sinon le branchement sera vers l'étiquette `testZeroRotation` (1 point).
- (h) Donner le code ARM complet du décalage de caractère dans le texte (ligne 14) (1 point).
- (i) Donner le code ARM partiel de la gestion de la boucle "pour" (ligne 10 et 22 seulement), le corps de la boucle sera remplacé par un commentaire `@ici corps de la boucle` (1.5 point).
- (j) Donner le code ARM complet de la boucle "tant que" (ligne 13, 14, 15 et 16) (2 points).

3 Du binaire au ternaire (6 points)

«Le système ternaire (ou trinaire) est le système de numération utilisant la base 3. Les chiffres ternaires sont connus sous le nom de trit (trinary digit), de manière analogue à bit.» source Wikipédia.

Pour cet exercice, nous rappelons les premières puissances de 2 et de 3, faites-en bon usage.

n	0	1	2	3	4	5	6	7	8
2^n	1	2	4	8	16	32	64	128	256
3^n	1	3	9	27	81	243	729	2187	6561

3.1 Base 3

La base 3 usuelle est similaire à la base 2 pour représenter les entiers naturels (\mathbb{N}), avec les adaptations suivantes : les chiffres autorisés sont 0, 1 et 2 ; ce sont les puissances de 3 qui sont utilisées ; l'algorithme des divisions successives s'applique en effectuant des divisions par 3 ; etc.

Questions (2 points) :

- (k) Donner la valeur décimale du nombre $(2022)_3$ représenté en base 3.
- (l) En base 3 combien de trits sont nécessaires pour écrire $(2022)_{10}$ représenté en base 10 ?
- (m) Écrire $(2022)_{10}$ en base 3.
- (n) Combien de trits sont nécessaires pour écrire les nombres naturels (\mathbb{N}) représentés en binaire (base 2) sur 32 bits ?

3.2 Complément à 3 (C3)

Le complément à 3 est similaire au complément à 2, il est défini pour représenter les entiers relatifs (\mathbb{Z}). Il repose sur les mêmes principes : il y a autant d'entiers positifs que d'entiers négatifs ; les entiers positifs sont représentés en base 3 usuelle ; pour obtenir un entier négatif, il faut prendre le complément à 3^n . Ainsi $X + (-X) = 3^n$ et pour un codage en base 3 sur n trits, cela donne $X + (-X) = 0$ ($-X$ est donc bien l'opposé de X).

Questions (2 points) :

- (o) Donner la valeur décimale du nombre $(2022)_{C_3}$ représenté en complément à 3 sur 4 trits.
- (p) Donner le codage de l'opposé de $(2022)_{C_3}$ représenté en complément à 3 sur 4 trits. (conseil : vérifier votre résultat en faisant l'addition $X + (-X)$)
- (q) Peut-on déterminer le signe d'un nombre représenté en complément à 3 sur 4 trits ? Si oui, comment ?

3.3 Notation ternaire équilibrée (3eq).

La notation ternaire équilibrée est une forme de numération de position issue de la base 3 qui utilise les chiffres -1, 0 et +1 au lieu de 0, 1 et 2. Ainsi la notation ternaire équilibrée permet de représenter les nombres négatifs. N. B. : -1 sera représenté par \ominus , 0 sera représenté par \odot et +1 sera représenté par \oplus .

Questions (2 points) :

- (r) Donner la valeur décimale du nombre $(\oplus\ominus\oplus\oplus)_{3eq}$ représenté en notation ternaire équilibrée sur 4 trits équilibrés.
- (s) Donner, si c'est possible, le codage de $(-1)_{10}$ en notation ternaire équilibrée sur 4 trits équilibrés.
- (t) Quel est l'intervalle des nombres relatifs représentés par la notation ternaire équilibrée sur 4 trits équilibrés ?
- (u) Peut-on déterminer le signe d'un nombre écrit en notation ternaire équilibrée sur 4 trits équilibrés ? Si oui, comment ?

4 Commentaires libres

- **Notes.** moyenne des 50 premières copies : 10 (notes entre 2 et 19)
- **Erreurs (habituelles).**
 - Confusion entre la représentation C2 et l'opération C2 (15% des copies, malgré l'indication en début de sujet).
 - Confusion entre mov (entre registres) et ldr (entre registre et mémoire)
 - Confusion entre ldr (lecture à partir de la mémoire) et str (écriture en mémoire)
 - Confusion entre b et bl
 - Erreur de taille, ldr vs ldrb, selon entier ou adresse = 32 bits = 4 octets = 1 mot et caractère = 1 octet = 8 bits
 - Chaînes : les espaces comptent, à la fin un caractère NULL est ajouté pour utiliser les fonctions d'affichages standards
 - Oubli des sauts inconditionnels en fin de boucle et entre "alors" et "sinon"
 - Oubli de l'initialisation et/ou de l'incrémentatation du compteur dans les boucles "pour"
- **Autres (erreurs).**
 - La première question donne des réponses de taille variée : entre 2 lignes et plus d'une page (50 lignes)

- Attention pour les conditions qui ne sont pas immédiatement liées à une comparaison : mauvais placement pour les étiquettes de début de boucle (sur le début de lecture `txt[i]` et non pas sur le `cmp`). Souvent, les boucles ont une condition simple et l'étiquette est sur le `cmp`, mais ce n'est pas toujours le cas.
- La vérification de la validité du C3 (opération) n'a pas été souvent effectuée (ou montrée) et paradoxalement elle n'a pas eu l'effet escompté, sur 5 confirmations, 2 seulement sont correctes (biais de confirmation ?)
- Confusion entre C3 et `3eq` (comme si `3eq` n'était qu'une représentation de C3)
- Orthographe : `bit` prend un `s` au pluriel : 8 bits (et pas 8 bit).
- **Conseils.**
 - Bien lire l'énoncé
 - Appliquer les motifs de traductions
 - Garder son bon-sens
 - Garder un œil critique
 - Appliquer des méthodes de contrôle et de vérification

ANNEXES

A Fonctions d'entrée/sortie

Nous rappelons les principales fonctions d'affichages du fichier `es.s` :

- `bl EcrNdecimal32` (resp. `EcrNdecimal16`, `EcrNdecimal8`) affiche le contenu de `r1` en décimal sous la forme d'un entier naturel de 32 bits (resp. 16 bits, 8 bits).
- `bl EcrChaine` affiche la chaîne de caractères dont l'adresse est dans `r1`.
- `bl ALaLigne` provoque un passage à la ligne dans l'affichage.

B Principales instructions du processeur ARM

Code	Nom	Explication du nom	Opération	Remarque
0000	AND	AND	et bit à bit	
0001	EOR	Exclusive OR	ou exclusif bit à bit	
0010	SUB	SUBstract	soustraction	
0011	RSB	Reverse SuBstract	soustraction inversée	
0100	ADD	ADDition	addition	
1000	TST	TeST	et bit à bit	pas <code>rd</code>
1001	TEQ	Test EQivalence	ou exclusif bit à bit	pas <code>rd</code>
1010	CMP	CoMPare	soustraction	pas <code>rd</code>
1011	CMN	CoMpare Not	addition	pas <code>rd</code>
1100	ORR	OR	ou bit à bit	
1101	MOV	MOVE	copie	pas <code>rn</code>
1110	BIC	BIt Clear	et not bit à bit	
1111	MVN	MoVe Not	not (complément à 1)	pas <code>rn</code>
	Bxx	Branch	branchement conditionnel	xx = condition Cf. table ci-dessous
	LDR	LoAD Register	lecture mémoire	
	STR	STore Register	écriture mémoire	

L'opérande source d'une instruction MOV peut être une valeur immédiate notée #5 ou un registre noté Ri, i désignant le numéro du registre. Il peut aussi être le contenu d'un registre sur lequel on applique un décalage de k bits; on note Ri, DEC #k, avec DEC ∈ {LSL, LSR, ASR, ROR}.

C Codes conditions du processeur ARM

Nous rappelons les codes de conditions arithmétiques xx pour l'instruction de branchement Bxx.

cond	mnémorique	signification	condition testée
0000	EQ	égal	Z
0001	NE	non égal	\bar{Z}
0010	CS/HS	≥ dans N	C
0011	CC/LO	< dans N	\bar{C}
0100	MI	moins	N
0101	PL	plus	\bar{N}
0110	VS	débordement	V
0111	VC	pas de débordement	\bar{V}
1000	HI	> dans N	$C \wedge \bar{Z}$
1001	LS	≤ dans N	$\bar{C} \vee Z$
1010	GE	≥ dans Z	$(N \wedge V) \vee (\bar{N} \wedge \bar{V})$
1011	LT	< dans Z	$(N \wedge \bar{V}) \vee (\bar{N} \wedge V)$
1100	GT	> dans Z	$\bar{Z} \wedge ((N \wedge V) \vee (\bar{N} \wedge \bar{V}))$
1101	LE	≤ dans Z	$Z \vee (N \wedge \bar{V}) \vee (\bar{N} \wedge V)$
1110	AL	toujours	