GIF-1001, 25 février 2014 Nom/Matricule : _____Solution_____

Examen 1

Cet examen vaut 40% de la note totale du cours. Les questions seront corrigées sur un total de 40 points. La valeur de chaque question est indiquée avec la question. Une calculatrice scientifique peut être utilisée. Cependant, aucune documentation, autre que l'annexe A, n'est permise. Vous pouvez répondre aux questions directement sur ce questionnaire et/ou dans le cahier bleu mis à votre disposition.

Q1 (2 points) : Quel mot ou concept relié aux ordinateurs correspond à la définition suivante :

#	Définition	Mot(s)
A	Circuit logique à l'intérieur du microprocesseur responsable des calculs arithmétiques.	*ALU*
B	Très petite mémoire (quelques octets) à l'intérieur du microprocesseur permettant d'entreposer des données temporairement pour effectuer des calculs.	*Registre*
C	Partie du microprocesseur responsable du décodage et de l'exécution des instructions	*Unité de contrôle centrale (CCU)*
D	Exécution en parallèle de plusieurs instructions, mais de parties différentes de chaque instruction exécutées.	*Pipeline*

Q2 (6 points) : On retrouve le code assembleur suivant à l'intérieur d'un ordinateur :

Adresse	Instruction	Binaire
0x80	MOV R0, #0	0xE3A00000
0x84	LDR R1, [R0]	0xE5901000
0x88	ADD R2, R1, R0	0xE0802001
0x8C	STR R2, [R0]	0xE5802000

Répondez aux questions qui suivent à partir du code ci-dessus. Toutes les questions sont indépendantes. S'il manque une information pour répondre, mettez une variable dans votre réponse. Par exemple, s'il vous faut savoir le contenu de la mémoire à l'adresse 0x96, indiquez quelque chose comme Memoire[0x96] dans votre réponse…

a) Dites brièvement, dans vos mots, ce que fait le programme.
b) À partir du binaire, quelle est la taille d'une instruction et quelle semble être la taille des opcodes (codes op)?
c) Si PC vaut 0x80 initialement, indiquez les valeurs, en fonction du temps, qui apparaîtront sur les bus d'adresse, de données et de contrôle lors de la lecture et l'exécution des quatre instructions.

a) Le programme met R0 = 0, puis il lit la mémoire à l'adresse 0. Enfin, il additionne 0 au contenu de ce qu'il y a à l'adresse 0 et remet le résultat dans la mémoire, à l'adresse 0. Le programme ne fait rien…
b) Entre 12 bits et 18bits.

c) Plusieurs points à comprendre avant de répondre :

- Le microprocesseur lit et exécute des instructions
- Les instructions sont dans la mémoire, sous forme de séquences bits. La séquence de bits représentant une instruction a deux parties : l'opcode –ce que doit exécuter le microprocesseur- et les paramètres –l'information que doit traiter le microprocesseur-.
- Les registres sont dans le microprocesseur. Un registre est une variable de 32 bits (équivaut à 32 bascules D), qui contient une valeur.
- La mémoire peut être vue comme une table ayant pour index l'adresse. À chaque adresse de mémoire, on retrouve 8 bits (1 byte) d'information.
- Il faut quatre bytes d'information (32 bits) pour décrire une instruction. Donc, les adresses mémoires progressent de 4 entre chaque instruction.
- Le microprocesseur contient le registre PC (Program Counter) qui dit l'adresse de l'instruction à lire et exécuter.
- Pour lire une donnée ou une instruction de la mémoire (ou d'un périphérique) :
    - Le microprocesseur met l'adresse à lire sur le bus d'adresse.
    - Le microprocesseur active la broche de contrôle « Lecture »
    - La mémoire qui est activée (en fonction de l'adresse) met un mot de mémoire sur le bus de données. Les bits mis sur le bus de donnée par la mémoire sont indiqués par l'adresse.
    - Le microprocesseur lit les bits provenant de la mémoire du bus de donnée.
- Pour écrire une donnée en mémoire (ou d'un périphérique) :
    - Le microprocesseur met l'adresse à écrire sur le bus d'adresse.
    - Le microprocesseur met les bits à écrire (désignés par l'instruction exécutée) sur le bus de données.
    - Le microprocesseur active la broche de contrôle « Écriture »
    - La mémoire ou le périphérique qui est activé (en fonction de l'adresse) lit le bus de données. Les bits lus du bus de donnée sont mis en mémoire à l'emplacement indiqué par l'adresse.
- L'instruction MOV Rn, #X dit au microprocesseur de mettre Rn = X. Rn et X sont des paramètres de l'instruction MOV.
- L'instruction LDR Rn, [adr] dit au microprocesseur de lire la mémoire à l'adresse indiquée par la valeur de *adr*. Puis, l'information lue est mise dans Rn.

Donc, la réponse à la question 3c est :

Lecture de MOV R0, #0	Le microprocesseur met 0x80 sur le bus d'adresse
	Le microprocesseur active « Lecture »
	La mémoire met 0xE3A00000 sur le bus de données
	Le microprocesseur lit l'instruction sur le bus de données
Exécution, MOV *R0*, #0	Le registre 0 (en italique) et la valeur 0 sont encodées dans l'instruction (en gras) 0xE3A0***0*****<u>000</u>**. R0, à l'intérieur du microprocesseur, est mis à 0. *Il ne se produit rien sur les bus lors de l'exécution de l'instruction.*
PC est incrémenté de 4 (la taille d'une instruction) par le microprocesseur	
Lecture de LDR *R1*, [R0]	Le microprocesseur met 0x84 sur le bus d'adresse
	Le microprocesseur active « Lecture »
	La mémoire met 0xE590***1***000 sur le bus de données
	Le microprocesseur lit l'instruction sur le bus de données
Exécution, LDR R1, [R0]	Le microprocesseur met la valeur de R0 (0!) sur le bus d'adresse
	Le microprocesseur active « Lecture »
	La mémoire met les bits à l'adresse 0 sur le bus de données. Disons que ces bits valent « Mem[0] »
	Le microprocesseur lit Mem[0] sur le bus de données. Ensuite, il met R1 = Mem[0], mais cela se fait à l'intérieur du microprocesseur.
PC est incrémenté de 4 (la taille d'une instruction) par le microprocesseur	
Lect. de ADD *R2*, **R1**, <u>R0</u>	Le microprocesseur met 0x88 sur le bus d'adresse
	Le microprocesseur active « Lecture »
	La mémoire met 0xE0802<u>0</u>0**1** sur le bus de données
	Le microprocesseur lit l'instruction sur le bus de données
Exéc. ADD R2, R1, R0	*Il ne se produit rien sur les bus lors de l'exécution de l'instruction.*
PC est incrémenté de 4 (la taille d'une instruction) par le microprocesseur	
Lecture de STR R2, [R0]	Le microprocesseur met 0x8C sur le bus d'adresse
	Le microprocesseur active « Lecture »
	La mémoire met 0x E5802000 sur le bus de données
	Le microprocesseur lit l'instruction sur le bus de données
Exécution, STR R2, [R0]	Le microprocesseur met la valeur de R0 (0) sur le bus d'adresse
	Le microprocesseur met la valeur de R2 (Mem[0]) sur le bus de données.
	Le microprocesseur active la ligne d'écriture
	Le mémoire lit le bus de données. Elle met la valeur lue (Mem[0]) à l'adresse désignée (R0 = 0)
PC est incrémenté de 4 (la taille d'une instruction) par le microprocesseur	
…	

Q3 (4 points) : Décrivez la hiérarchie de mémoire de l'ordinateur. En d'autres mots, indiquez ce qui se produit lorsque le microprocesseur veut exécuter du code qui se trouve sur le disque dur seulement.

Lorsque l'ordinateur exécute des instructions, il regarde d'abord dans les caches (L1 en premier, L3 en dernier) si les instructions s'y trouvent. Si elles ne sont pas là, l'ordinateur regarde dans la mémoire. Enfin, si les instructions ne se trouvent pas en mémoire, le système les lit sur le disque dur.

Quand une donnée ou une instruction est lue sur le disque dur, un bloc de donnée (la donnée et celles adjacentes) est copié en mémoire pour diminuer le temps d'accès moyen à l'information, en vertu du principe de localité. De la même manière une plus petite partie des données sera recopiée dans les caches.

Q4 (2 points) : Pour toutes les questions qui suivent, choisissez une réponse unique parmi les choix.

A) Comment le microprocesseur sait-il à quelle adresse finit la mémoire d'instruction?
   a. Le microprocesseur a un registre qui contient l'adresse de fin de la mémoire d'instruction. Lorsque PC dépasse cette adresse, PC devient 0.
   b. L'adresse de la fin de la mémoire d'instruction est une constante initialisée dans le microprocesseur lors de la production en usine.
   c. Une instruction spéciale à la fin du programme indique au microprocesseur que les instructions sont terminées.
   d. Le microprocesseur ne sait pas à quelle adresse finit la mémoire d'instruction.

B) Comment le microprocesseur sait-il s'il manipule des données ou des instructions en mémoire?
   a. Les adresses réservées aux instructions et celles réservées aux données sont des constantes dans le microprocesseur et ce dernier se base sur ces constantes pour savoir.
   b. Un bit de chaque mot de mémoire indique si le mot de mémoire contient une instruction ou une donnée.
   c. Des registres du microprocesseur lui disent s'il accède à une donnée ou à une instruction
   d. Le microprocesseur ne sait pas si les adresses qu'il utilise désignent des données ou des instructions.

C) Comment la mémoire sait-elle la plage d'adresses qui lui est réservée dans l'ordinateur pour se connecter au bus de donnée uniquement lorsque requis?
   a. Des registres à l'intérieur de la mémoire permettent d'établir l'adresse de base de la mémoire.

b. Des broches de la mémoire connectées à des interrupteurs permettent de déterminer les adresses de la mémoire.
c. Une ligne du décodeur d'adresse active la mémoire avec une broche Enable lorsque l'adresse provenant du microprocesseur désigne la mémoire.
d. Il n'y a pas de plage d'adresses réservée à chaque mémoire.

D) Dans un système MMIO (Memory mapped Input/Output, comme le coeur ARM), comment le microprocesseur sait-il s'il accède à la mémoire ou à un périphérique?
a. Des registres à l'intérieur du microprocesseur disent au microprocesseur les adresses de la mémoire et celles des périphériques.
b. Les instructions exécutées par le microprocesseur lui disent s'il accède à la mémoire ou aux périphériques
c. Ce sont les périphériques qui indiquent au microprocesseur les adresses permettant d'y accéder.
d. Le microprocesseur ne sait pas s'il accède à la mémoire ou aux périphériques.

Q5 (7 points) : Sachant que tous les nombres du système ordinateur sont en Little Endian (petit boutisme), écrivez une fonction en assembleur ARM 32 bits qui additionne deux nombres ayant 64 bits. Cette fonction reçoit trois paramètres d'entrée : les adresses des nombres à additionner et l'adresse du résultat. L'adresse des nombres à additionner est dans R0 et R1. Le résultat de l'addition doit être mis à l'adresse désignée par R2 : MEM[R2] = MEM[R0] + MEM[R1].

Tenez compte des éléments suivants dans votre réponse :
- Une liste d'instructions ARM se retrouve en Annexe A
- Vous aurez besoin de l'instruction ADC pour répondre correctement
- Tous les registres doivent avoir la même valeur après l'appel de la fonction qu'avant l'appel de la fonction.
- Votre fonction doit être appelée comme suit :

Appel de la fonction	Code de la fonction
LDR R0, =Nombre_A_Additonner1 LDR R1, =Nombre_A_Additonner2 LDR R2, =Resultat BL Addition64bits ;suite du programme ici	Addition64bits: *PUSH {R3,R4,R5,R6} ;Sauvegarde Regs* *LDR R3, [R0] ;Lecture LSB* *LDR R4, [R1] ;Lecture LSB* *ADDS R5, R3, R4 ;Add LSB, Set flags* *LDR R3, [R0,#4] ;Lecture MSB* *LDR R4, [R0,#4] ;Lecture MSB* *ADDC R6, R3, R4 ;Add avec C* *STR R5,[R2] ;Sauvegarde resultat LSB* *STR R6,[R2,#4] ;Sauvegarde resultat MSB* *POP {R3, R4, R5,R6} ;Restore Regs* *BX LR ;Retour de fonction*

b) Décrivez comment on pourrait utiliser la pile plutôt que les registres pour passer les paramètres d'entrée (les nombres à additionner) et le paramètre de retour (le résultat).

Appel de la fonction	Code de la fonction
LDR R0, =Nombre_A_Additonner1 LDR R1, =Nombre_A_Additonner2 LDR R2, =Resultat ***PUSH {R0, R1, R2}*** BL Addition64bits ;suite du programme ici	Addition64bits: *PUSH {R0-R6} ;Sauvegarde Regs* ***LDR R0, [SP,#40]*** ***LDR R1, [SP,#36]*** ***LDR R2, [SP,#32]*** *LDR R3, [R0] ;Lecture LSB* *LDR R4, [R1] ;Lecture LSB* *ADDS R5, R3, R4 ;Add LSB, Set flags* *LDR R3, [R0,#4] ;Lecture MSB* *LDR R4, [R0,#4] ;Lecture MSB* *ADDC R6, R3, R4 ;Add avec C* *STR R5,[R2] ;Sauvegarde resultat LSB* *STR R6,[R2,#4] ;Sauvegarde resultat MSB* *POP {R0-R6} ;Restore Regs* *BX LR ;Retour de fonction*

Q6 (2 points) : Classez ces mémoires de la plus rapide à la moins rapide : SRAM, DRAM, FLASH, Mémoire magnétique. Parmi ces quatre mémoires, quelles mémoires sont volatiles? en lecture seule?

Rapide : SRAM (plus rapide) – DRAM – FLASH – Mémoire magnétique (moins rapide)
Volatiles : SRAM et DRAM
Lecture seule : FLASH (peut être écrite, mais l'opération est longue et complexe)

Q7 (2 points) : Effectuez les opérations mathématiques suivantes. ***Considérez que tous les nombres sont sur 8bits en notation complément 2*** :

Opérande 1	123	0xF0	10001010b	00110011
Opération	+	-	ET	OU
Opérande 2	12	0x0B	11110101b	11001000
Résultat	*-121*	*0xE5*	*10000000b*	*11111011b*

Q8 (5 points) : Décrivez le comportement matériel et logiciel du système lorsqu'un périphérique active une ligne d'interruption. En d'autres mots, dites quels circuits électroniques interviendront dans le traitement de l'interruption ainsi que leurs rôles. Décrivez également l'effet de l'interruption sur l'exécution du programme : comment l'ordinateur entrera-t-il dans le traitement de l'interruption et que fera-t-il lorsque le traitement sera terminé?

Matériel : Le périphérique active un signal d'interruption relié au contrôleur d'interruption. Le contrôleur d'interruption active une broche d'interruption du microprocesseur qui négocie l'interruption avec le contrôleur d'interruption.

Entrée dans l'interruption :

- *Le microprocesseur finit le ou les instructions en cours*
- *Le microprocesseur regarde si l'interruption peut être exécutée.*
- *Le microprocesseur empile l'information nécessaire à la reprise de l'exécution sur la pile (PC, drapeaux, registre de lien, et plus)*
- *Le microprocesseur cherche l'adresse de l'ISR à être exécutée pour traiter l'information.*
- *Le microprocesseur met PC au début de l'ISR et commence l'exécution*

Sortie de l'interruption :

- *Le microprocesseur rencontre et exécute une instruction qui lui dit de revenir de l'interruption*
- *Le microprocesseur dépile l'information l'information nécessaire à la reprise de l'exécution de la pile*
- *Le microprocesseur reprend l'exécution à partir de l'endroit où il était avant l'interruption.*

Q9 (5 points) : Dites si les énoncés suivants sont vrais ou faux (0.5 par bonne réponse, 0 si réponse absente et -0.25 par mauvaise réponse.

#	Énoncé	V/F
4A	Lorsqu'une fraction est représentée sur 32 bits avec la norme IEEE754, 19 bits représentent la partie entière avec le signe et 13 bits représentent la fraction (Valeur/2^{13}).	*F*
B	Lorsque vous représentez du texte avec la table ASCII, le texte prend moins d'espace en mémoire que lorsque vous le faites avec l'Unicode.	*V*
C	Le DMA (Direct Memory Access) permet d'écrire des données de la mémoire vers un périphérique ou vice versa sans intervention du microprocesseur.	*V*
D	Une mémoire dynamique est une mémoire qui peut être écrite ou lue à plusieurs adresses simultanément.	*F*
E	Le mot de contrôle est un groupe de bits produit par le microprocesseur afin de réaliser une partie d'une instruction.	*V*
F	Un microprocesseur avec pipeline d'instruction exécutera toujours au moins une instruction par coup d'horloge en exécutant les instructions en parallèle.	*F*
G	La pile de l'ordinateur est située dans la mémoire RAM même si le pointeur de pile est un registre situé dans le microprocesseur.	*V*
H	Un bit du mot de contrôle indique si un registre contient une valeur ou une adresse.	*F*
I	Le microprocesseur ARM ne supporte pas d'instruction spécifique pour faire des rotations ou des décalages de bits parce qu'un circuit électronique spécialisé, le barrel shifter le fait dans le microprocesseur.	*V*
J	Les constantes (exemple : PI) et les variables sont gérées de la même manière dans un système ordinateur avec microprocesseur ARM sauf qu'elles ne sont pas dans les mêmes mémoires	*V* *

Q10 (3 points) : Expliquez comment l'instruction Branch est utilisée pour faire des énoncés conditionnels en assembleur. Convertissez, an assembleur la condition « si a > 8, ExécuteTacheVrai(), sinon, ExecuteTacheFaux() »… Vous retrouverez du support sur l'instruction B et les codes de condition en annexe.

```
        LDR   R0, =a
        CMP   R0,a
        BHI   TacheVrai
        ;ExecuteTacheFaux()
        B     FinDeCondition
TacheVrai :
        ;ExecuteTacheVrai()
FinDeCondition :
```

Q11 (2 points) : Classez les interruptions suivantes de la plus prioritaire à la moins prioritaire. En cas de doute, expliquez votre réponse : Reset, Exécution d'instruction invalide, interruption logicielle, interruption non-masquable (NMI), interruption du clavier et interruption de la souris.

Reset – NMI – Instruction invalide – Interruption logiciel – clavier ou souris selon les instructions du programmeur.

QA (Bonus : 1 points) : Habituellement, les programmes se terminent par une interruption. Pourquoi?
Pour retourner au système d'exploitation.

QB (Bonus : 1 points) : Que se passe-t-il lorsque PC dépasse la dernière adresse de la mémoire d'instruction?
Le microprocesseur lit et exécute des instructions.
Il lira instructions sur le bus de donnée qui proviennent de nulle part, ce qui conduira probablement à une exception.

Annexe A : Instructions ARM et Codes de condition

Mnemonic	Description
ADD Rd, Rs, Op1	Rd = Rs + Op1
ADC Rd, Rs, Op1	Rd = Rs + Op1 + C
AND Rd, Rs, Op1	Rd = Rs AND Op1
B{cc} Offset	PC = PC + Offset, si cc est rencontré
BL{cc} Offset	Comme B, LR = Adr. de l'instr suivante
CMP Rs, Op1	Change les drapeaux comme Rs-Op1
LDR Rd, [Rs, Op2]	Rd = Mem[Rs + Op2]
LDMdd {Reg List}, [Rs, Op2]	Load Multiple Registers
MUL Rd, Rs, Op1	Rd = Rs * Op1
MVN Rd, Op1	Rd = !Op1
POP {Reg List}	Met la liste de registres sur la pile
PUSH {Reg List}	Met la liste de registres sur la pile
SBC Rd, Rs, Op1	Rd = Rs - Op1 - C
STMdd {Reg List}, [Rs, Op2]	Store Multiple Registers
STR Rd, [Rs, Op2]	Mem[Rs + Op2] = Rd
SUB Rd, Rs, Op1	Rd = Rs - Op1

Codes de Conditions				
Mnemonic	Condition		Mnemonic	Condition
CS	Carry Set		CC	Carry Clear
EQ	Equal (Zero Set)		NE	Not Equal (Zero Clear)
VS	Overflow Set		VC	Overflow Clear
GT	Greater Than		LT	Less Than
GE	Greater Than or Equal		LE	Less Than or Equal
PL	Plus (Positive)		MI	Minus (Negative)
HI	Higher Than		LO	Lower Than (aka CC)
HS	Higher os Same (aka CS)		LS	Lower or Same