

Microprocesseurs 1 & 2: Travail écrit no 1.

Nom : CASTELLETTI

Prénom : ALESSIO

Classe : I/2

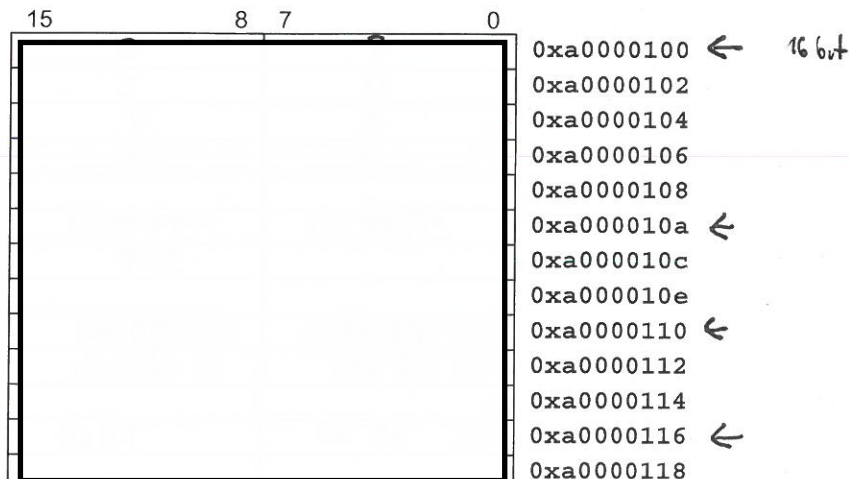
Date : 21.11.2011

Problème n° 1 (architecture générale)

a) Dessinez et décrivez succinctement l'architecture Von Neumann:

b) Pour une organisation de la mémoire est en « little-endian », représentez (en hexadécimal pour les entiers et en caractère ascii pour les strings) dans le tableau ci-dessous les variables suivantes :

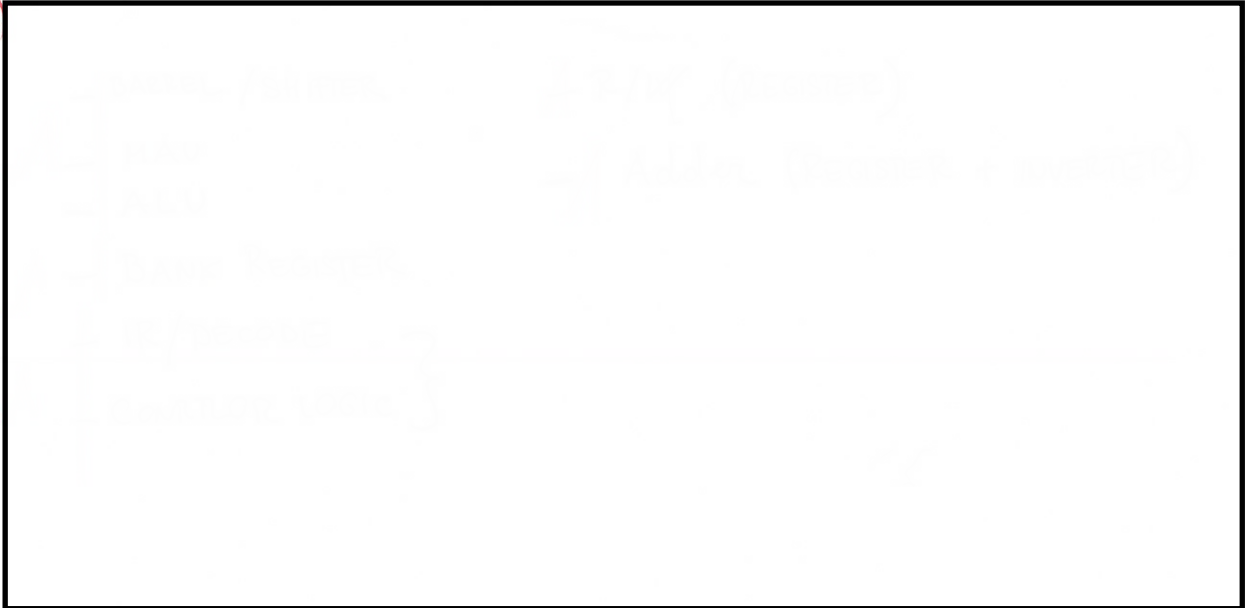
Adresse :	variable :	taille/type :	valeur :
0xa0000100	var1	.asciz	"bonjour"
0xa000010d	var2	.byte 8	252 <sub>10</sub>
0xa0000110	var3	.long 32	67832b <sub>16</sub>
0xa0000116	var4	.short 16	406 <sub>8</sub>
0xa000010a	var5	.byte 8	-4 <sub>10</sub>



Microprocesseurs 1 & 2: Travail écrit no 1.

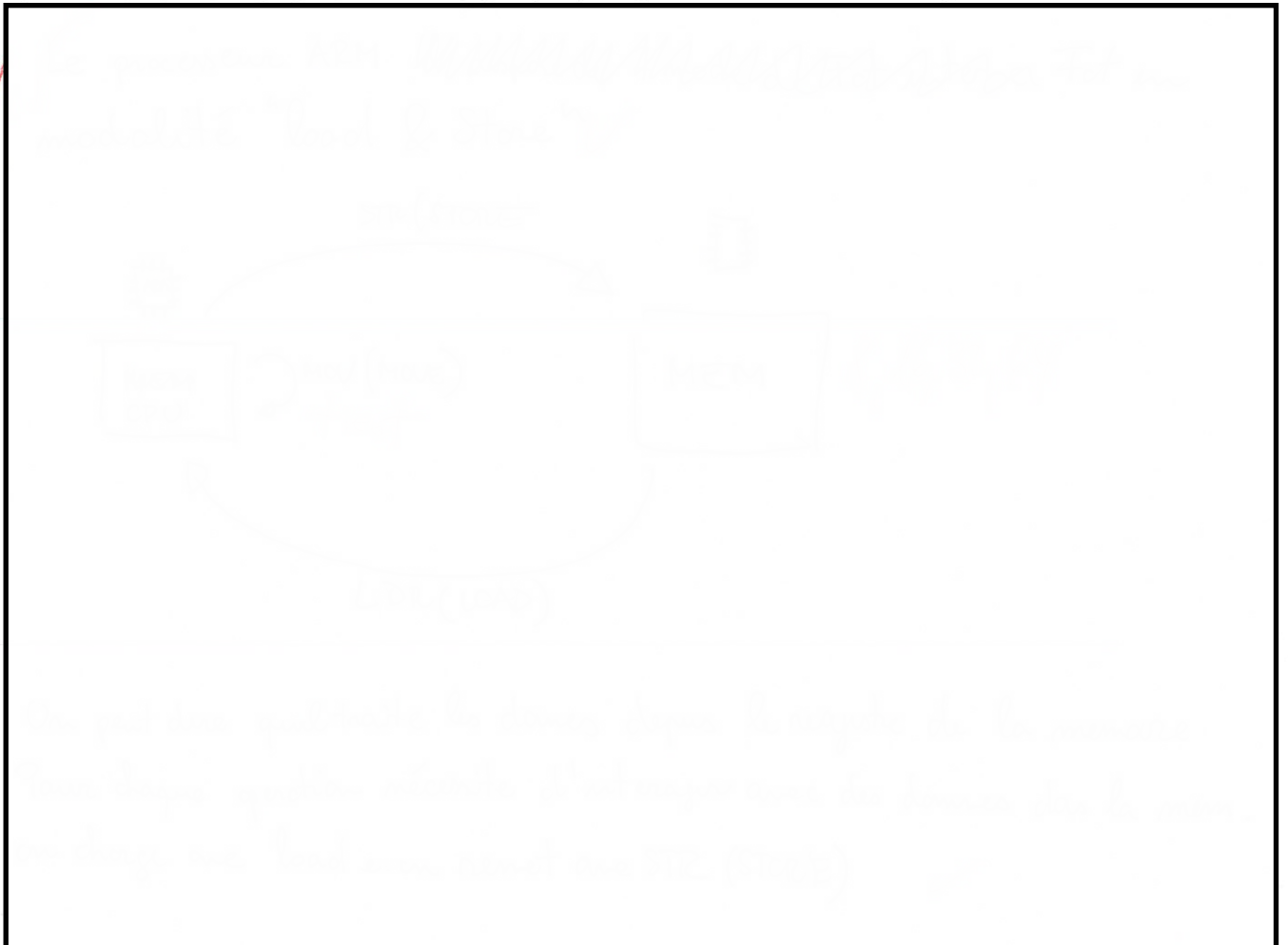
10 **Problème n° 2** (architecture interne)

- a) Citez ou dessinez les composants principaux de la structure interne des processeurs ARM :



- b) Décrivez (avec un graphique ou une figure) le principe de fonctionnement des processeurs ARM:

4



Microprocesseurs 1 & 2: Travail écrit no 1.

9 Problème n° 3 (traitement numérique des nombres)

- a) Prévoyez l'état des flags Z, C, N et V ainsi que le résultat contenu dans le registre R0 (en décimal) suite à l'exécution des instructions assembleur suivantes :

*Remarque : toutes les opérations sont faites avec des registres de 8 bits au lieu de 32 bits*

1. ldr r0, #247

2 cmp r0, #-9

2. mov r0, #139

2 adds r0, #193

3. mov r0, #-3

2 subs r0, #125

- b) Représentez en hexadécimal sur 32 bits (simple précision) les valeurs réelles suivantes et donner le développement :

(pour rappel : exposant est codé sur 8 bits avec un biais de 127)

1.5 a) 45,375 :

1.1 b) 49 / 2048 :

$\times 2^{-11}$

Microprocesseurs 1 & 2: Travail écrit no 1.

Problème n° 4 (Mode d'adressage)

Pour le code assembleur et la représentation de la mémoire (little-endian / 8-bits) et l'état des registres du processeur ci-dessous, donnez le résultat des opérations (état des registres, état de la mémoire):

Mémoire  
(little-endian / 8 bits)

0xa0001000	0x43
0xa0001001	0x83
0xa0001002	0x97
0xa0001003	0x25
0xa0001004	0xd7
→ 0xa0001005	0x25
0xa0001006	0x73
0xa0001007	0xc2
0xa0001008	0xaa
0xa0001009	0x89
0xa000100a	0x00
0xa000100b	0xc0
→ 0xa000100c	
0xa000100d	
0xa000100e	
0xa000100f	

Registres  
(avant)

R0	0xa000'0100
R1	0x0000'1022
R2	0x0000'0400
R3	0xffff'ff00
R4	0xa000'1000
R5	0x0000'0001
R6	0x0000'0004
R7	0xffff'8ff6
R8	0xa000'1008
R9	0x0000'0100
R10	0x0000'0000
R11	0xa000'0100
R12	0x0000'0010

Registres  
(après)

0. 0xa000'4400: backup: .long 102,105,106,107,110,111,112

1. add r0,r1,r2,lsl #4

2. ldrb r3, [r4,r6]

3. strh r7, [r4,#12]!

4. ldr r1, [r8],r12,lsl #4

0x a000 1008

0x a000 1008

5. ldr r9, =backup  
ldmia r9!, {r2,r5-r7,r10-r12}

Microprocesseurs 1 & 2: Travail écrit no 1.

2 Problème n° 5 (Programmation en assembleur)

Coder en langage assembleur ARM l'algorithme suivant :

① { #define MAX 200  
char str[] = "un message avec des chiffres 123458";  
char msg[MAX]; long digits=0; short len=0;  
  
void main() {  
int i = 0; int j=0;  
do {  
char c = str[i++];  
if ((c >= '0') && (c <= '9')) {  
digits++; c = '\*';  
}  
msg[j++] = c;  
len++;  
} while ((c != 0) && (len < MAX));  
msg[MAX-1] = 0;  
}  
---en assembleur-----  
MAX = 200  
str: .asciz "un message avec des chiffres 123458"  
msg: .space MAX  
digits: .long 0  
len: .short 0  
  
main: