

**GIF-1001 Ordinateurs: Structure et Applications**  
**Hiver 2018**  
**Examen final**  
**1er mai 2018**  
**Durée : 180 minutes**  
**Professeur : Jean-François Lalonde**

---

Cet examen comporte 8 questions sur 14 pages (incluant celle-ci), comptabilisées sur un total de 100 points. L'examen compte pour 40% de la note totale pour la session.

- Vous avez droit à une feuille aide-mémoire  $8.5 \times 11$  recto-verso, écrite à la main, ainsi qu'une calculatrice acceptée.
- Écrivez vos réponses dans le cahier bleu qui vous a été remis ;
- Veuillez placer une carte d'identité sur votre table pour vérification ;
- Nous vous fournissons quatre (4) annexes :
  - l'annexe A (p. 11) contient des rappels sur la conversion d'unités et sur les logarithmes ;
  - l'annexe B (p. 12) contient une liste d'instructions ARM ainsi que des codes de conditions ;
  - l'annexe C (p. 13) contient la liste des registres banqués en assembleur ARM ;
  - l'annexe D (p. 14) contient la table ASCII.

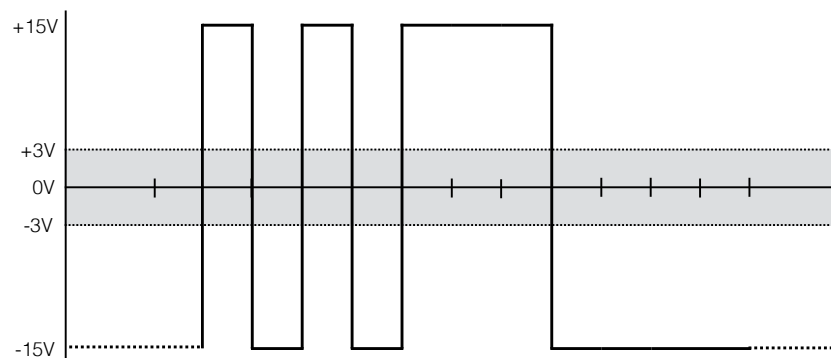
La table ci-dessous indique la distribution des points pour chaque question.

Question:	1	2	3	4	5	6	7	8	Total
Points:	15	10	15	10	10	15	15	10	100

Bonne chance et bon été!

1. (15 points) Un système communique avec un périphérique via une connexion série qui emploie le protocole RS-232, et qui possède les caractéristiques suivantes :
- 8 bits par mot ;
  - parité impaire ;
  - 1 bit d'arrêt ;
  - vitesse de transfert de 19 200 bits par seconde.

(a) Le système envoie le signal illustré dans la figure ci-bas.



- i. (3 points) Sachant que la plage entre +3V et +15V correspond à un «0» logique, et que la plage entre -3V et -15V à un «1» logique, convertissez le signal en bits. Identifiez clairement : le bit de départ («start bit»), le bit de parité, ainsi que le bit d'arrêt («stop bit»).
  - ii. (2 points) Quel est le mot reçu en hexadécimal ?
- (b) Le système envoie ensuite la série de bits suivante sur la ligne : 0-1-0-1-1-0-0-1-0-1-1. De l'autre côté, le périphérique détermine qu'il a reçu le mot 0x6E.
- i. (2 points) Y a-t-il eu erreur de transmission ? Si oui, indiquez le mot que le périphérique aurait dû recevoir.
  - ii. (3 points) Si le bit de parité reçu par le périphérique est «1», est-ce que le périphérique peut savoir s'il y a eu erreur de transmission ? Pourquoi ?
- (c) (3 points) On désire envoyer la chaîne de caractères «OSA» sur la ligne. Chaque caractère est encodé en ASCII (voir annexe D). Quelle sera la chaîne de bits à envoyer sur la ligne de transmission ?
- (d) (2 points) Le roman «Les Misérables» de Victor Hugo contient environ 513 000 mots. Si on suppose une moyenne conservatrice de 4 lettres par mot et que chaque mot est suivi d'un espace ou d'un signe de ponctuation, ce chef-d'oeuvre contiendrait environ 2 565 000 caractères.

On veut transmettre «Les Misérables» sur le lien série décrit ci-haut en encodant chaque caractère en ASCII (voir annexe D). Cependant, le lien est soumis à de fortes perturbations électromagnétiques, ce qui fait que 2% des caractères reçus sont erronés et doivent être retransmis. Vous pouvez assumer que les caractères erronés sont toujours détectés et simplement envoyés deux fois.

Combien de temps ce système prendra-t-il pour transférer «Les Misérables» sur un tel lien série ? Donnez votre réponse en minutes.

2. (10 points) Les processus suivants sont admis, dans l'ordre.

Nom du processus	Durée (quantum)	Arrivée (quantum)	Priorité
P1	5	0	haute
P2	2	1	basse
P3	2	3	haute
P4	3	4	basse

Indiquez quel processus sera exécuté à chaque quantum de temps pour chacun des algorithmes spécifiés ci-bas.

*Important* : si deux processus sont équivalents (par exemple, deux processus ayant le même temps restant pour l'algorithme «plus court d'abord»), choisissez celui *arrivé en premier*.

- (a) (1 point) Premier arrivé, premier servi (sans tenir compte de la priorité).
- (b) (1 point) Plus prioritaire d'abord. Si deux (ou plus) processus ont la même priorité, utilisez le premier arrivé, premier servi.
- (c) (2 points) Plus court d'abord (sans tenir compte de la priorité).
- (d) (2 points) Plus prioritaire d'abord. Si deux (ou plus) processus ont la même priorité, utilisez le plus court d'abord.
- (e) (2 points) Tourniquet (sans tenir compte de la priorité).
- (f) (2 points) Plus prioritaire d'abord. Si deux (ou plus) processus ont la même priorité, utilisez le tourniquet. Dans ce cas, vous devez maintenir une file d'attente différente pour chaque niveau de priorité.

3. (15 points) Dans un système avec mémoire paginée où :
- 17 bits sont nécessaires pour représenter la position d'une adresse dans une page ;
  - la taille de la mémoire virtuelle est de 2Go ;
  - la taille de la mémoire physique est de 256Mo ;
  - un extrait de la table des pages est donné par :

Page	Trame
0x00	0x01
0x01	0xB2
0x02	0xCD
0x03	0x05
0x04	0x9F
0x05	0x32
0x06	0x2D
0x07	0x7C
0x08	0x11
0x09	0x09
0x0A	0xBD
⋮	⋮

- (a) (2 points) Quel est le nombre de pages dans ce système ?
- (b) (2 points) Quel est le nombre de trames dans ce système ?
- (c) (4 points) Si la table des pages ne stocke que le numéro de trame pour chaque page, quelle est la taille totale de la table des pages ? Écrivez votre réponse en kilo-octets (Ko).
- (d) (4 points) Traduisez l'adresse virtuelle 0x154236 en adresse physique en utilisant la table des pages ci-haut. Écrivez clairement votre démarche.
- (e) (2 points) À quelle *page* l'adresse *physique* 0x5A3CF9 correspond-elle ? Écrivez clairement votre démarche.
- (f) (1 point) Combien y a-t-il d'éléments dans la table des pages *inversée* de ce système ?

4. (10 points) La liste suivante indique des processus en attente d'allocation mémoire (dans l'ordre de leur arrivée), ainsi que leur taille. La taille totale de la mémoire physique du système est de 64Ko.
1. P1, 7Ko
  2. P2, 2Ko
  3. P3, 10Ko
  4. P4, 8Ko
  5. P5, 5Ko

Commençons tout d'abord par analyser le cas d'une architecture utilisant la stratégie d'allocation par mémoire contigüe avec partitions de taille *fixe*. Les partitions ont une taille de 8Ko.

- (a) (2 points) Est-ce que le système parvient à allouer de l'espace à tous les processus ? Sinon, quel processus ne peut être chargé ?
- (b) (2 points) Quelle sera la fragmentation interne totale lorsque tous les processus pouvant être chargés seront en mémoire ?

Analysons maintenant le cas d'un autre système, toujours à mémoire contigüe, mais qui utilise cette fois des partitions de taille *variable*. Les mêmes processus ci-haut sont tout d'abord admis en mémoire. Ensuite, les processus P1 et P3 se terminent. Finalement, les nouveaux processus suivants surviennent, dans l'ordre :

1. P6, 8Ko
  2. P7, 2Ko
- (c) (3 points) Écrivez le contenu de la mémoire si l'algorithme suivant est utilisé pour allouer de l'espace aux nouveaux processus : meilleure allocation («best fit») ?
  - (d) (3 points) Écrivez le contenu de la mémoire si l'algorithme suivant est utilisé pour allouer de l'espace aux nouveaux processus : prochaine allocation («next fit») ?

5. (10 points) En 2019, les responsables du cours GIF-1001 ont bonifié le simulateur ARM pour simuler les communications avec les périphériques. Comme vous avez bien réussi le cours en 2018, vous êtes engagé(e) comme dépanneur(e). Votre première tâche est de concevoir un nouveau TP qui permet de simuler les interactions avec un scanner, soit la numérisation d'une page, et le transfert de cette page numérisée vers la RAM.

Vous écrivez le code suivant pour remplir une partie de la tâche :

```
1 LDR R0, =AdresseScanner
2 LDR R3, =AdresseEnRAM
3 LDR R1, NombreOctetsACopier
4
5 PUSH {R5}
6 MOV R2, #0
7 boucle
8 LDR R5, [R0], #4
9 STR R5, [R3], #4
10 ADD R2, R2, #1
11
12 CMP R2, R1
13 BNE boucle
14 POP {R5}
```

- (a) (2 points) Décrivez, *sans écrire de code*, une alternative à la stratégie employée ci-haut pour communiquer avec le scanner. Détaillez votre réponse.
- (b) (3 points) Nommez un avantage et un inconvénient de la solution que vous avez proposée en (a) par rapport à la stratégie employée par le code ci-haut ?

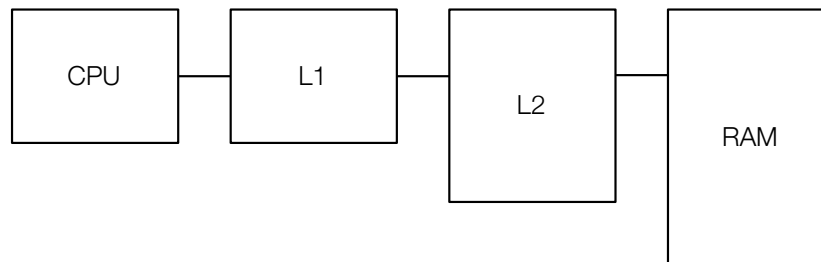
Vous écrivez cet autre code pour remplir une autre partie de la tâche :

```
1 ; Demande au scanner de numeriser une page.
2 ; Pour ce faire, on lui envoie le code "0xAB"
3 LDR R0, =ControleScanner
4 MOV R3, #0xAB
5 STR R3, [R0]
6
7 LDR R0, =EtatScanner
8 boucle
9 LDR R3, [R0]
10 CMP R3, #1 ; si le bit 0 est a 1, le scanner a termine
11 BNE boucle
```

- (c) (2 points) Décrivez, *sans écrire de code*, une alternative à la stratégie employée ci-haut pour communiquer avec le scanner. Détaillez votre réponse.
- (d) (3 points) Nommez un avantage et un inconvénient de la solution que vous avez proposée en (c) par rapport à la stratégie employée par le code ci-haut ?

6. (15 points) Un système possède les caractéristiques suivantes :
- deux caches (L1 et L2) ;
  - les blocs en cache contiennent 8 mots ;
  - la cache L1 :
    - est de type «write-back» ;
    - ne possède aucun bloc vide ;
    - possède un bloc le moins récemment utilisé qui *n'est pas* sale ;
  - la cache L2 :
    - est de type «write-through» ;
    - ne possède aucun bloc vide ;

Ce système est illustré dans la figure suivante :



Répondez aux questions suivantes portant sur ce système.

- (a) (3 points) Décrivez les étapes nécessaires à l'écriture d'un mot d'un bloc qui est présent en cache L1.
- (b) (4 points) Décrivez les étapes nécessaires à l'écriture d'un mot d'un bloc qui est présent en cache L2, mais pas en cache L1.
- (c) (2 points) Si la cache L2 était «write-back» plutôt que «write-through», qu'aurait-il fallu faire différemment pour l'écriture d'un mot d'un bloc présent en cache L2 mais pas en L1, comme en (b) ?
- (d) Sachant que le temps de transfert
- d'un *mot* de la cache L1 vers le CPU est de 4ns ;
  - d'un *bloc* de la cache L2 vers la cache L1 est de 64ns ;
  - d'un *bloc* de la RAM vers la cache L2 est de 256ns ;
- i. (2 points) Quel temps sera nécessaire pour accéder aux 8 mots d'un même bloc qui n'est pas en cache ?
  - ii. (2 points) Quel temps sera nécessaire pour accéder à 8 mots répartis sur 4 blocs qui sont en cache L2 mais pas en cache L1 ?
  - iii. (2 points) Quel temps sera nécessaire pour accéder à 4 mots répartis sur 4 blocs qui sont tous en cache L1 ?

7. (15 points) Toujours dans le cadre de votre mandat de dépanneur(e) du cours GIF-1001 à l'hiver 2019 (voir question 5), vous vous voyez confier la tâche d'assister les étudiants durant une période de dépannage au PLT-0103. Pour vous préparer à votre période de dépannage, vous révisez les notes de cours qui vous rappellent que :

Adresse	Interruption
0x00	Reset
0x04	Instruction indéfinie
0x08	Interruption logicielle
0x0C	«Prefetch abort»
0x10	«Data abort»
0x14	Espace réservé
0x18	IRQ
0x1C	FIQ

Lors d'un TP sur les interruptions, un étudiant vient vous voir car son code ne fonctionne pas. Le code de l'étudiant est disponible sur la page suivante. Après l'avoir analysé, vous parvenez à trouver cinq (5) erreurs.

- (3 points) Vous décelez une erreur dans la table des vecteurs d'interruption. Décrivez l'erreur, identifiez clairement la ou les lignes concernées, et proposez une solution pour la régler.
- (3 points) Vous décelez une erreur dans le programme principal (lignes 13–24). Décrivez l'erreur, identifiez clairement la ou les lignes concernées, et proposez une solution pour la régler.
- (3 points) Vous décelez une erreur dans la routine de traitement de l'interruption SVC. Décrivez l'erreur, identifiez clairement la ou les lignes concernées, et proposez une solution pour la régler.
- (3 points) Vous décelez une autre erreur dans la routine de traitement de l'interruption SVC. Décrivez l'erreur, identifiez clairement la ou les lignes concernées, et proposez une solution pour la régler.
- (3 points) Vous décelez *deux* erreurs dans la routine de traitement de l'interruption FIQ. Décrivez *une de ces deux erreurs* (celle de votre choix), identifiez clairement la ou les lignes concernées, et proposez une solution pour la régler.



Voici le code de l'étudiant :

```
1 SECTION INTVEC
2
3 B main          ; interruption reset
4 NOP
5 B interruptionSVC ; interruption logicielle
6 NOP
7 MOV R0, #0
8 NOP
9 B interruptionFIQ ; interruption FIQ
10
11 SECTION CODE
12
13 main          ; Debut du code principal
14
15 MOV R0, #0
16 SVC #10      ; Appel systeme pour demarrer une requete au disque dur
17
18 boucleAttente
19 CMP R0, #0
20 B boucleAttente
21
22 ; Le code se continue...
23 fin
24 B fin
25
26 interruptionSVC
27
28 LDR R13, =ControleDisqueDur
29 MOV R14, #0x1
30 STR R14, [R13] ; envoyer la requete au disque dur
31
32 SUBS PC, LR, #-4 ; fin de la routine de traitement de l'interruption SVC
33
34 interruptionFIQ
35
36 ; charger les bonnes adresses
37 LDR R8, =AdresseDisqueDur
38 LDR R9, =AdresseEnRAM
39 LDR R10, NombreOctetsACopier
40
41 B copieMemoire ; copier la memoire
42 MOV R0, #1     ; terminer la boucle
43
44 BX LR         ; fin de la routine de traitement de l'interruption IRQ
45
46 copieMemoire
47 ; cette fonction est ecrite sans erreur ici...
48 BX LR
```

8. (10 points) Répondez aux questions suivantes par une réponse courte.
- (a) (1 point) Lorsqu'il y a une erreur de transmission en communication série, comment peut-on faire pour savoir quel bit est fautif ?
  - (b) (1 point) Quel est le rôle principal du MMU ?
  - (c) (1 point) En mémoire paginée, quelle information est stockée dans la table des pages *inversée* ?
  - (d) (1 point) Pourquoi y a-t-il deux lignes dédiées au transport des données dans le bus USB ?
  - (e) (1 point) Vrai ou faux ? Après être dans l'état «bloqué», un processus devient dans l'état «en cours» dès que la requête du périphérique est terminée.
  - (f) (1 point) Pourquoi l'ordre d'exécution de deux boucles imbriquées est-il important ?
  - (g) (1 point) À quoi sert un contrôleur de périphérique ?
  - (h) (1 point) Vrai ou faux ? La mémoire virtuelle doit avoir la même taille que la mémoire physique.
  - (i) (1 point) Le bus USB ne supporte pas les interruptions. De quelle façon sont-elles simulées ?
  - (j) (1 point) Nommez un inconvénient d'une cache «write-back» par rapport à une cache «write-through» ?

## A Annexe : Unités et logarithmes

### A.1 Unités

Petit rappel sur les unités :

$$\begin{aligned} 1\text{Ko} &= 2^{10} \text{ octets} &= & 1\,024 \text{ octets} \\ 1\text{Mo} &= 2^{20} \text{ octets} &= 1\,024\text{Ko} &= 1\,048\,576 \text{ octets} \\ 1\text{Go} &= 2^{30} \text{ octets} &= 1\,024\text{Mo} &= 1\,073\,741\,824 \text{ octets} \end{aligned}$$

### A.2 Logarithme en base 2

Pour calculer des logarithmes en base 2 à partir de logarithmes dans une autre base  $N$  (ex : 10), appliquez l'équation suivante :

$$\log_2 x = \frac{\log_N x}{\log_N 2}.$$

## B Annexe : Instructions ARM et codes de conditions

Instruction	Description
ADD Rd, Rs, Op1	$Rd \leftarrow Rs + Op1$
AND Rd, Rs, Op1	$Rd \leftarrow Rs \text{ AND } Op1$
ASR Rd, Rs, #cte	$Rd \leftarrow Rs / 2^{cte}$
B etiquette	$PC \leftarrow \text{adresse}(\text{etiquette})$
BL etiquette	$LR \leftarrow PC - 4, PC \leftarrow \text{adresse}(\text{etiquette})$
BX Rs	$PC \leftarrow Rs$
CMP Rs, Op1	Change les drapeaux comme $Rs - Op1$
LDR Rd, etiquette	$Rd \leftarrow \text{valeur}(\text{etiquette})$
LDR Rd, =etiquette	$Rd \leftarrow \text{adresse}(\text{etiquette})$
LDR Rd, [Rb, Op1]	$Rd \leftarrow \text{Mem}[Rb + Op1]$
LDR Rd, [Rb], Op1	$Rd \leftarrow \text{Mem}[Rb], Rb \leftarrow Rb + Op1$
LDR Rd, [Rb, Op1]!	$Rb \leftarrow Rb + Op1, Rd \leftarrow \text{Mem}[Rb]$
LSL Rd, Rb, #cte	$Rd \leftarrow Rb \times 2^{cte}$
MUL Rd, Rn, Rs	$Rd \leftarrow Rn \times Rs$
MVN Rd, Op1	$Rd \leftarrow !Op1$ (inverse les bits)
POP {Liste Reg}	Charge les registres en ordre croissant à partir de la pile, $SP \leftarrow SP - 4 \times (\text{nombre de registres})$
PUSH {Liste Reg}	$SP \leftarrow SP + 4 \times (\text{nombre de registres})$ , Met la liste de registres sur la pile dans l'ordre décroissant
STR Rs, etiquette	$\text{valeur}(\text{etiquette}) \leftarrow Rs$
STR Rs, [Rb, Op1]	$\text{Mem}[Rb + Op1] \leftarrow Rs$
STR Rs, [Rb], Op1	$\text{Mem}[Rb] \leftarrow Rs, Rb \leftarrow Rb + Op1$
STR Rs, [Rb, Op1]!	$Rb \leftarrow Rb + Op1, \text{Mem}[Rb] \leftarrow Rs$
SUB Rd, Rs, Op1	$Rd \leftarrow Rs - Op1$

TABLE 1 – Instructions ARM. Op1 dénote une opérande de type 1, soit une constante, un registre ou un registre décalé.

Code	Condition	Code	Condition
CS	Retenue (carry)	CC	Pas de retenue
EQ	Égalité	NE	Inégalité
VS	Débordement	VC	Pas de débordement
GT	Plus grand	LT	Plus petit
GE	Plus grand ou égal	LE	Plus petit ou égal
PL	Positif	MI	Négatif

TABLE 2 – Codes de condition.



**D Annexe : Table ASCII**

Dec	Hx	Oct	Char	Dec	Hx	Oct	Char	Dec	Hx	Oct	Char
0	0	000	NUL	43	2B	053	+	86	56	126	V
1	1	001	SOH	44	2C	054	,	87	57	127	W
2	2	002	STX	45	2D	055	-	88	58	130	X
3	3	003	ETX	46	2E	056	.	89	59	131	Y
4	4	004	EOT	47	2F	057	/	90	5A	132	Z
5	5	005	ENQ	48	30	060	0	91	5B	133	[
6	6	006	ACK	49	31	061	1	92	5C	134	\
7	7	007	BEL	50	32	062	2	93	5D	135	]
8	8	010	BS	51	33	063	3	94	5E	136	^
9	9	011	TAB	52	34	064	4	95	5F	137	_
10	A	012	LF	53	35	065	5	96	60	140	'
11	B	013	VT	54	36	066	6	97	61	141	a
12	C	014	FF	55	37	067	7	98	62	142	b
13	D	015	CR	56	38	070	8	99	63	143	c
14	E	016	SO	57	39	071	9	100	64	144	d
15	F	017	SI	58	3A	072	:	101	65	145	e
16	10	020	DLE	59	3B	073	;	102	66	146	f
17	11	021	DC1	60	3C	074	<	103	67	147	g
18	12	022	DC2	61	3D	075	=	104	68	150	h
19	13	023	DC3	62	3E	076	>	105	69	151	i
20	14	024	DC4	63	3F	077	?	106	6A	152	j
21	15	025	NAK	64	40	100	@	107	6B	153	k
22	16	026	SYN	65	41	101	A	108	6C	154	l
23	17	027	ETB	66	42	102	B	109	6D	155	m
24	18	030	CAN	67	43	103	C	110	6E	156	n
25	19	031	EM	68	44	104	D	111	6F	157	o
26	1A	032	SUB	69	45	105	E	112	70	160	p
27	1B	033	ESC	70	46	106	F	113	71	161	q
28	1C	034	FS	71	47	107	G	114	72	162	r
29	1D	035	GS	72	48	110	H	115	73	163	s
30	1E	036	RS	73	49	111	I	116	74	164	t
31	1F	037	US	74	4A	112	J	117	75	165	u
32	20	040	Space	75	4B	113	K	118	76	166	v
33	21	041	!	76	4C	114	L	119	77	167	w
34	22	042	"	77	4D	115	M	120	78	170	x
35	23	043	#	78	4E	116	N	121	79	171	y
36	24	044	\$	79	4F	117	O	122	7A	172	z
37	25	045	%	80	50	120	P	123	7B	173	{
38	26	046	&	81	51	121	Q	124	7C	174	
39	27	047	'	82	52	122	R	125	7D	175	}
40	28	050	(	83	53	123	S	126	7E	176	~
41	29	051	)	84	54	124	T	127	7F	177	DEL
42	2A	052	*	85	55	125	U				