GIF-1001, 1 mars 2011 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, 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) : Décrivez ce que fait le microprocesseur de votre ordinateur lors d'une interruption, d'un point de vue logiciel. En d'autres mots, dites quelles actions sont entreprises automatiquement par le cœur de votre ordinateur quand un périphérique signale un évènement. Pour vous guider, la première action exécutée par le microprocesseur est indiquée ci-dessous:

0) Une interruption se produit.
1) Le microprocesseur termine adéquatement la ou les instructions en cours.

2) Le microprocesseur vérifie s'il peut exécuter l'interruption.
3) Le microprocesseur sauvegarde l'adresse de l'instruction à exécuter quand l'interruption sera traitée.
4) Le microprocesseur trouve l'adresse de la sous-routine qui traitera l'interruption à un endroit prédéterminé.
5) Le microprocesseur fait un saut vers la sous-routine traitant l'interruption.

Q2a (2 points) : Énumérez trois usages de la pile gérée par le microprocesseur de votre ordinateur. À quoi sert cette structure de donnée en mémoire?
Q3b (1 point bonus) Trouvez deux autres utilisations de la pile.

1- *Passer des paramètres à une fonction*
2- *Sauvegarder les adresses de retour d'une fonction (lors d'un CALL ou lors d'une interruption)*
3- *Sauvegarder des valeurs de registres ou variables pour les récupérer plus tard (rendre une fonction propre par exemple)*
4- *Utiliser la pile pour entreposer les variables locales*
5- *Utiliser la pile pour effectuer certaines opérations mathématiques (exemple : swap de registres, chaînes de calculs mathématiques, valider que toutes les parenthèses d'une chaîne de caractère ferment…)*

Q4 (5 points) : L'instruction *MOV mémoire, mémoire* n'est pas supportée par le 8086 : elle est remplacée par deux instructions : *MOV registre, mémoire* suivie de *MOV mémoire, registre*. Montrez que l'instruction MOV mémoire, mémoire est plus complexe que l'instruction MOV registre, mémoire en détaillant toutes les µ-instructions (micro-instructions ou sous-instructions) qui doit réaliser le microprocesseur lorsqu'il lit et exécute chacune de ces instructions.

a) Décrivez les µ-instructions pour lire et exécuter *MOV registre, mémoire*
b) Décrivez les µ-instructions pour lire et exécuter *MOV mémoire, mémoire*

****

MOV registre, mémoire ; lecture de l'instruction

- *Mettre le compteur de programme sur le bus d'adresse*
- *Activer la ligne de lecture de la mémoire*
- *Attendre que la mémoire mette l'instruction sur le bus de données*
- *Lire le bus de données et mettre la valeur lue dans le registre d'instruction*
- *Incrémenter le compteur de programme de la taille de l'instruction*

MOV registre, mémoire ; exécution de l'instruction

- *Mettre l'adresse de mémoire désignée dans l'instruction sur le bus d'adresse*
- *Activer la ligne de lecture de la mémoire*
- *Attendre que la mémoire mette la donnée sur le bus de données*
- *Lire le bus de données et mettre la valeur lue dans le registre désigné par l'instruction*

****

MOV mémoire, mémoire ; lecture de l'instruction

- *Mettre le compteur de programme sur le bus d'adresse*
- *Activer la ligne de lecture de la mémoire*
- *Attendre que la mémoire mette l'instruction sur le bus de données*
- *Lire le bus de données et mettre la valeur lue dans le registre d'instruction*
- *Incrémenter le compteur de programme de la taille de l'instruction*

MOV mémoire1, mémoire2 ; exécution de l'instruction

- *Mettre l'adresse de mémoire2 désignée dans l'instruction sur le bus d'adresse*
- *Activer la ligne de lecture de la mémoire*
- *Attendre que la mémoire mette la donnée sur le bus de données*
- *Lire le bus de données et mettre la valeur lue dans un buffer temporaire*
- *Mettre l'adresse de mémoire1 désignée dans l'instruction sur le bus d'adresse*
- *Mettre la donnée du buffer temporaire sur le bus de données*
- *Activer la ligne d'écriture de la mémoire*
- *Attendre que la mémoire lise la donnée sur le bus de données*

Q5 (2 points) : La plupart des adresses de destination des sauts conditionnels ou inconditionnels sont relatives à l'adresse de l'instruction de saut. Par exemple, vous retrouverez JMP +8 ou JZ -20 plutôt que JMP 0x00320. Quels sont les avantages des sauts relatifs par rapport aux sauts absolus?

- *L'instruction est plus courte avec des sauts relatifs*
- *Le code se déplace plus facilement avec des sauts relatifs*

Q6 (2 points) : Pourquoi les instructions de saut conditionnel du 8086 sont-elles généralement précédées d'une instruction comme CMP ou TEST?

Pour changer/établir la valeur des drapeaux de l'ALU avant l'instruction de saut conditionnel du 8086 qui utilise les drapeaux de l'ALU afin de déterminer s'il faut faire le saut.

Q7 (6 points) : Écrivez une fonction en assembleur 8086 qui retourne la position du '1' le plus significatif d'un nombre non-signé, sur 16 bits, écrit en binaire. La position du '1' le plus significatif (celui valant le plus, le plus à gauche) est maximum 15 et minimum 0.

Exemple : Si le nombre est 00010000 01011111, votre fonction doit retourner 12, car le '1' le plus significatif, valant 2^{12}, est à la position 12.

Votre fonction, appelée par CALL, doit recevoir un nombre sur 16 bits en entrée. Elle doit retourner un nombre également sur 16 bits, même si le résultat de votre fonction est entre 0 et 15. La méthode de passage des paramètres est laissée à votre discrétion.

Si le nombre en entrée ne contient pas de 1 (s'il vaut 0), votre fonction doit retourner 0xFFFF.

Enfin, votre fonction doit être propre et elle doit être commentée (aucun commentaire = 0).

Note : Trouver le '1' le plus significatif d'un nombre en binaire est équivalent à calculer la partie entière du logarithme en base 2 de ce nombre!

Voici un exemple de solution, plusieurs solutions et approches étant possibles!!!

```
        org     100h
        jmp     start

Entree  DW    0x4001
Sortie  DW    0

start:
        PUSH Entree
        CALL TrouvePremierUn
        POP Sortie
        JMP start


TrouvePremierUn   PROC
        PUSH AX                 ;Sauvegarde des registres modifiés par la fonction
        PUSH BX
        PUSH CX
        PUSH BP
        MOV BP, SP

        MOV AX, [BP + 10]   ;Met le parametre d'entree dans AX
        CMP AX,0
        JNZ EntreeNonNulle
        MOV BX, 0xFFFF      ;Retourne 0xFFFF quand Entree == 0
        JMP Fin

EntreeNonNulle:

        MOV BX,15           ;BX contient la position du 1 le plus significatif
        MOV CX, 0x8000

ProchainUn:

        TEST AX,CX              ;Teste (ET) les bits de gauche à droite pour trouver 1
        JNZ Fin
        DEC BX
        SAR CX, 1
        JMP ProchainUn

Fin:
        MOV [BP+10], BX     ;Met le resultat sur la pile
        POP BP                  ;Récupération des registres modifies par la fonction
        POP CX
        POP BX
        POP AX
        RET
TrouvePremierUn ENDP
```

Q8 (6 points) : Le registre IP d'un 8086 vaut 0x0200 et le 8086 lira et exécutera les trois instructions suivantes :

Adresse de l'instruction	Instruction
0x10200	MOV BX, 3
0x10202	PUSH BX
0x10204	ADD [BX], AX

Sachant qu'initialement :

- CS, DS, ES et SS valent respectivement 0x1000, 0x2000, 0x3000 et 0x4000
- AX, BX, CX et DX valent respectivement 1, 2, 3, 4

Décrivez la séquence des valeurs qui apparaîtront sur le bus de données (16 bits), d'adresse (20 bits) et de contrôle (assumez deux lignes : lecture de mémoire et écriture de mémoire) du 8086 lors de l'exécution de cette séquence d'instructions.

Les variables dont la valeur initiale n'est pas donnée dans l'énoncé du problème doivent rester des variables, avec des noms explicites, à l'intérieur de votre réponse.

Par exemple, votre réponse pourrait commencer par :

Lecture de l'instruction MOV BX, 3	*Le microprocesseur met 0x10200 sur le bus d'adresse*
	Le microprocesseur active la ligne de lecture de la mémoire
	La mémoire met l'instruction MOV BX, 3 sur le bus de données
Exécution de MOV BX, 3	*Rien n'apparaît sur les bus d'adresse, de contrôle ou de données : le microprocesseur met 3 (contenu dans l'instruction) dans le registre BX qui est à l'intérieur du microprocesseur*
Lecture de l'instruction PUSH BX	*Le microprocesseur met 0x10202 sur le bus d'adresse*
	Le microprocesseur active la ligne de lecture de la mémoire
	La mémoire met l'instruction PUSH BX sur le bus de données
Exécution de PUSH BX	*Le microprocesseur met l'adresse du dessus de la pile sur le bus d'adresse.*
	Le microprocesseur met la valeur de BX (3) sur le bus de données
	Le microprocesseur active la ligne d'écriture de la mémoire
Lecture de l'instruction ADD [BX], AX	*Le microprocesseur met 0x10204 sur le bus d'adresse*
	Le microprocesseur active la ligne de lecture de la mémoire
	La mémoire met l'instruction ADD [BX], AX sur le bus de données
Exécution de l'instruction ADD [BX], AX	*Le microprocesseur met la valeur de BX (3) sur le bus d'adresse. Cette valeur est combinée avec DS (DS :BX = 0x20003) pour 20 bits.*
	Le microprocesseur active la ligne de lecture de la mémoire
	La mémoire met X (ce qu'il y a à l'adresse 0x20003) sur le bus de données
	Le microprocesseur met X+1 (AX vaut 1) sur le bus de données
	Le microprocesseur active la ligne d'écriture de la mémoire

Q9 (10 points) : Indiquez si les énoncés suivant sont vrai ou faux. Une bonne réponse vaut 1 point, une mauvaise réponse vaut -0.5 points et pas de réponse vaut 0.

#	Énoncé	V/F
A	Selon la norme IEEE754, les fractions peuvent être représentées avec 32 bits. Ainsi, selon ce standard, sur 32 bits, il est possible de représenter une fraction égale à $1/2^{31}$.	*V*
B	Supposez deux nombres, A et B, non-signés, sur 8 bits. Soustraire ces deux nombres donnera toujours la différence (non-signée) entre ces deux nombres, même si on effectue A-B et que B est supérieur à A.	*V-F*
	La réponse dépend de votre interprétation de A-B…	
C	Un système d'exploitation est un programme qui interprète, à l'aide d'un programme d'interface usager, des commandes provenant de l'utilisateur afin d'exécuter d'autres programmes.	*V*
D	Lorsqu'un programme ne respecte pas le principe de localité temporelle, les données contenues dans la mémoire de votre ordinateur seront souvent remplacées par des données provenant du disque dur.	*V*
E	Lorsqu'un programme ne respecte pas le principe de localité spatiale, les données contenues dans les caches de votre ordinateur seront souvent remplacées par des données provenant de la mémoire.	*V*
F	Le chipset est un circuit intégré qui fait le lien entre la carte-mère de votre ordinateur et le microprocesseur.	*V*
G	Le DMA permet de transférer des données automatiquement de la mémoire vers un périphérique en soulageant le microprocesseur de cette tâche. C'est la mémoire qui déclenche le transfert par DMA.	*F*
H	Un bit de chaque instruction ou variable est géré par l'assembleur ou le compilateur afin d'indiquer au microprocesseur si la valeur à l'adresse en cours désigne une instruction ou une variable.	*F*
I	Dans le 8086 on s'aperçoit que les variables et les instructions sont mélangées dans la même mémoire ce qui est conforme à l'architecture de Von Neumann.	*V*
J	Le registre Program Counter (PC ou IP dans le 8086) ne peut être modifié par aucune instruction : en effet, c'est le microprocesseur qui l'incrémente automatiquement à la fin de chaque instruction.	*F*

Q10 (3 points) Un disque dur tourne à 6000 RPM (révolution par minute) et prend, en moyenne vingt (20) millisecondes (ms) pour trouver et lire un bloc de données. Sachant que ce disque dur a 10 secteurs, 6 têtes de lecture et 100 pistes par surface, déterminez quel est le temps moyen de déplacement radial d'une tête de lecture vers le bloc de données à lire.

Temps moyen recherche et lecture = Temps moyen de déplacement de la tête (X)
+ Temps pour faire un demi-tour (1/2/vitesse)
*+ Temps pour lire le bloc (1/[vitesse*Nsecteurs])*

20ms = X + 5ms + 1ms -> X = 14ms.

Q11 (2 points) À l'aide d'un exemple ou en reformulant, expliquez ce que signifie l'affirmation suivante : « Les directives d'assembleur ifdef, ifndef et define servent à modifier un programme avant la compilation lorsque le programme sert dans différents environnements logiciels ou matériels ».

Supposons un logiciel qui contrôle le déplacement d'un ascenseur de type A et qui pourrait contrôler également un ascenseur de type B, si on lui apporte des modifications mineures. Alors, les directives ifdef, ifndef et define pourraient/devraient changer le logiciel avant la compilation afin de tenir compte du modèle d'ascenseur.

QBonus (2 points bonus) : Expliquer pourquoi les interruptions du BIOS sont de moins en moins utilisées dans vos applications.

Les interruptions du BIOS servent pour accéder aux périphériques de base de votre ordinateur. Le rôle des interruptions se limite à contrôler quelques périphériques clefs afin permettre le chargement et l'exécution du système d'exploitation. Comme les périphériques sont de plus en plus complexes et variés, le BIOS en supporte de moins en moins et c'est le système d'exploitation qui gère de plus en plus les interruptions permettant d'accéder aux périphériques.

Annexe A : Liste non exhaustive des instructions du 8086

Instruction	Description
ADD a,b	Effectue a = a+b.
AND a,b	Effectue a = a ET b, où ET est un ET logique.
CALL proc	Appelle la procédure proc et empile l'emplacement de retour.
CLC	Met à 0 le drapeau de retenue (carry).
CMP a,b	Effectue a-b, a et b sont inchangés.
DEC a	Décrémente a.
DIV mot	Effectue AX = DXAX/mot, non signé, le résultat est tronqué (arrondi inférieur).
IDIV mot	Effectue AX = DXAX/mot, signé, le résultat est tronqué (arrondi inférieur).
IMUL mot	Effectue DXAX = AX*mot, signé.
IN dst, port	Met la valeur lue sur le port de I/O port dans dst.
INC a	Incrémente a.
INT a	Appelle la routine de service d'interruption a. Empile les drapeaux et l'emplacement de retour.
IRET	Retourne d'une int. en dépilant l'emplacement de retour et les drapeaux.
JC label	Saute à l'instruction désignée par label si le drapeau Carry est 1.
JMP label	Saute à label. La prochaine instruction exécutée est désignée par label.
JNC label	Saute à l'instruction désignée par label si le drapeau Carry est 0.
JNZ label	Saute à l'instruction désignée par label si le drapeau Zéro est 0.
JNS label	Saute à l'instruction désignée par label si le drapeau Signe est 0.
JZ label	Saute à l'instruction désignée par label si le drapeau Zéro est 1.
JS label	Saute à l'instruction désignée par label si le drapeau Signe est 1.
LEA dst,var	Met l'adresse de la variable var dans dst.
MOV dst,src	Met le contenu de src dans dst. Ne change pas les drapeaux.
MUL octet	Effectue AX = AL*octet, non signé.
NEG a	Inverse tous les bits de a, puis ajoute 1.
NOT a	Inverse tous les bits de a.
OR a,b	Effectue a = a OU b, où OU est un OU logique.
OUT port, src	Met la valeur de src sur le port de I/O port.
POP mot	Dépile un mot. Ne change pas les drapeaux.
POPF	Dépile les drapeaux.
PUSH mot	Empile un mot. Ne change pas les drapeaux.
PUSHF	Empile les drapeaux.
RET et RETF	Retourne d'une procédure en dépilant l'emplacement de retour. RET dépile IP. RETF dépile IP puis CS.
RCL a,b	Fait une rotation de b bits vers la gauche. La rotation inclut le bit de Carry.
RCR a,b	Fait une rotation de b bits vers la droite. La rotation inclut le bit de Carry.
SAL a,b	Décale tous les bits de a vers la gauche d'un nombre de bits égal à b. Des zéros sont mis à droite. Carry prend la valeur du bit disparu.
SAR a,b	Décale tous les bits de a vers la droite d'un nombre de bits égal à b. Le bit le plus significatif de a est mis à gauche. Carry prend la valeur du bit disparu.
STC	Met à 1 le drapeau de Carry
SUB a,b	Effectue a = a-b.
TEST a,b	Effectue a ET b, où ET est un ET logique.
XOR a,b	Effectue a = a XOR b, où XOR est un OU eXclusif.