EXERCICES DIRIGES 4

LANGAGE MACHINE

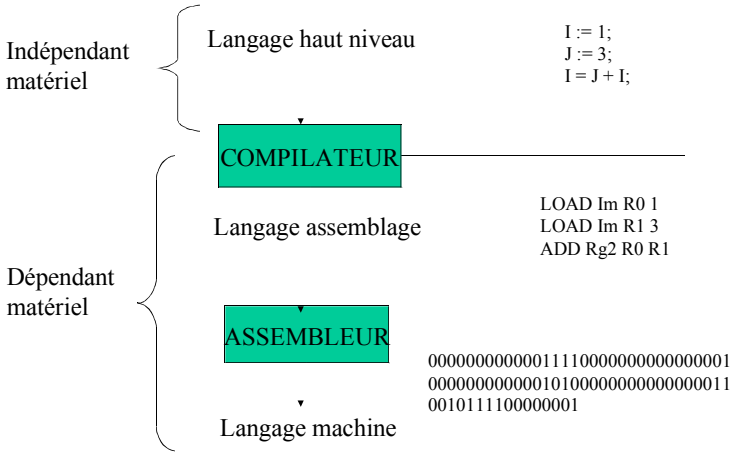
Rappels de cours

1. Notion d'instructions, modes d'adressages

Le langage de programmation d'une machine est un ensemble d'**instructions** permettant de spécifier à la machines les opérations à exécuter pour résoudre un problème donné. Il existe différents niveaux de langages.
Au niveau de la machine physique se trouve le langage dit langage machine, seul exécutable par la machine. Les instructions du langage machine sont des mots binaires.

Le schéma ci-dessous résume les différents niveaux de langages existants sur une machine :

- le langage de haut niveau (C, ada, Cobol, etc…) est indépendant de la machine physique et ne peut pas être directement exécuté par celle-ci.
- le langage d'assemblage est dépendant de la machine. Il est équivalent au langage machine à la différence près que les chaines binaires de l'instruction machine sont remplacées par des chaines mnémoniques.
- le langage machine est composé des instructions directement exécutées par le processeur.



On se réfère à la documentation MAP32 pour étudier le format des instructions et les différents types d'instructions.

Spécification MAP32

I. La mémoire

Elle est constituée de mots de 32 bits.
La mémoire est accédée par le biais de deux registres processeur, le registre RAD (Registre Adresse) et le registre RDO (Registre Données), chacun de ces registres étant de 32 bits.
Une zone de la mémoire est gérée comme une structure de pile.

II. L'unité centrale

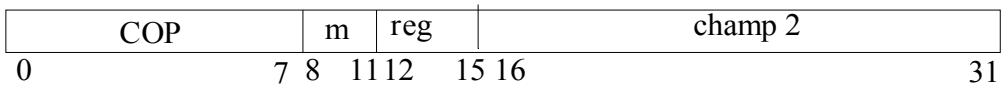
Les registres de l'unité centrale sont des registres de 32 bits.
On distingue :

- un compteur ordinal CO, qui contient l'adresse de la prochaine instruction à exécuter
- un registre instruction RI, qui contient l'instruction couramment exécutée
- 4 registres généraux : de R0 à R3
- 1 registre spécialisé pour l'adressage :
    - RB, le registre de base
- 1 registre Pointeur de pile : RSP
- les registres RAD et RDO
- 1 registre d'état, PSW, qui contient notamment les flags suivants :
    - O : positionné à 1 si Overflow, 0 sinon
    - Z : positionné à 1 si résultat opération nul, 0 sinon
    - C : positionné à 1 si carry, 0 sinon
    - S : positionné à 0 si résultat opération positif, 1 sinon
    - I : masquage des interruptions : positionné à 1 si interruption masquée, 0 sinon.

L'unité centrale contient également une Unité Arithmétique et Logique.

III. Format des instructions machine

La machine admet des instructions sur 32 bits.
Le format d'une instruction 32 bits est la suivante :



où

COP est le Code OPération codé sur 8 bits
m est le mode d'adressage codé sur 4 bits
reg code un numéro de registre sur 4 bits (de 0000 à 1111)
champ2 est une valeur immédiate, une adresse mémoire, un déplacement codé sur 16 bits ou un numéro de registre.

Les modes d'adressage; le champ m

La machine supporte les modes d'adressage mémoire suivants :

Immédiat	Opérande = valeur immédiate	Im	m = 0000
Direct	Opérande = [adresse]	D	m = 0001
Indirect	Opérande = [[adresse]]	I	m = 0010
Basé	Opérande = [[RB] + déplacement]	B	m = 0011

Pour ces valeurs de m comprises entre 0 et 3, le code opération travaille sur deux opérandes; le premier est un registre dont le numéro est codé par le champ reg; le deuxième est soit une valeur immédiate, soit une adresse déduite de champ2 et m.

Les valeurs m = 0100 et m = 0101 sont utilisées pour les opérations sur des registres :

0100 (Rg1) : le code opération travaille sur un seul opérande registre reg.

0101 (Rg2) : le code opération travaille sur deux registres l'un codé dans reg, l'autre dans champ2.

Les valeurs des registres

La valeur reg = 0000 à 0011 code les numéros de registres généraux R0 à R3. Les autres valeurs sont réservées pour coder les autres registres du processeur. Ainsi :

reg = 1110 registre RB

reg = 1111 registre PSW

Ces valeurs de registres sont également utilisées dans champ2 avec un mode d'adressage Rg2

Les instructions

Mnémonique	Code binaire
LOAD	00000000
STORE	00000001
PUSH	00000010
POP	00000011
ADD	00000100
MUL	00000101
NEG	00000110
AND	00000111
OR	00001000
XOR	00001001
NOT	00001010
JMP	00001011
JMPP	0001100
JMPO	00001101
JMPN	00001110
JMPC	00001111

JMPZ	00010000

Les instructions de transfert de données

Ces instructions permettent
- le transfert d'une donnée entre un registre banalisé et un emplacement mémoire
- le chargement d'un registre banalisé avec une valeur immédiate ou le contenu d'un autre registre.

Elles ne modifient pas le registre d'état, PSW.

- **Transfert d'un mot mémoire vers un registre banalisé**

LOAD m reg champ2	m = B, D, I, Rg2, Im	champ2 est une adresse ou un déplacement ou un numéro de registre ou une valeur immédiate.

Exemples :
LOAD D R1 $(000A)_{16}$: chargement du registre R1 avec la case mémoire d'adresse $(0000000A)_{16}$ adressée en mode direct.

LOAD Im R1 $(000A)_{16}$: chargement du registre R1 avec la valeur immédiate $(0000000A)_{16}$

LOAD Rg2 R1 R2 : chargement du registre R1 avec le contenu du registre R2

- **Transfert d'un registre vers un mot mémoire**

STORE m reg champ2	m = B, D, I	champ2 est une adresse ou un déplacement

Exemples :
STORE D R1 $(000A)_{16}$: écriture du contenu du registre R1 dans la case mémoire d'adresse $(000A)_{16}$ adressée en mode direct.

Les instructions de traitement des données

Ces instructions regroupent les fonctions mathématiques et les fonctions booléennes. Dans ces instructions, le registre d'état, PSW, est modifié en fonction du résultat de l'opération.

- **Fonctions arithmétiques**

Nous utiliserons essentiellement l'opération d'addition et de complémentation à 2.

ADD m reg champ2	m = B, D, I, Rg2, Im champ2 est soit une valeur immédiate, soit un opérande mémoire, soit un déplacement, soit un numéro de registre	addition entre le contenu de reg et l'opérande déduit de m et champ2, puis stockage du résultat dans reg
MUL m reg champ2	m = B, D, I, Rg2, Im champ2 est soit une valeur immédiate, soit un opérande mémoire, soit un déplacement, soit un numéro de registre	multiplication entre le contenu de reg et l'opérande déduit de m et champ2, puis stockage du résultat dans reg
NEG Rg1 reg		Complément à 2 de reg puis stockage du résultat dans reg.

Exemples :
ADD Im R0 $(000A)_{16}$: addition de la valeur immédiate $(000A)_{16}$ avec le contenu du registre R0 et stockage du résultat dans R0.

- **Fonctions booléennes**

AND m reg champ2 OR m reg champ2 XOR m reg champ2	m = B, D, I, Rg2, Im champ2 est soit une valeur immédiate, soit un opérande mémoire, soit un déplacement, soit un numéro de registre	ET logique (OU, OU exclusif) entre le contenu de reg et l'opérande déduit de m et champ2, puis stockage du résultat dans reg
NOT Rg1 reg		Complément à 1 de reg puis stockage du résultat dans reg.

Exemples :
AND Im R0 $(000A)_{16}$: ET logique entre la valeur immédiate $(000A)_{16}$ et le contenu du registre R0 et stockage du résultat dans R0.

Les instructions non séquentielles

Elles appartiennent à trois catégories : aucune ne modifie le registre d'état.

- sauts inconditionnels`

- sauts conditionnels

- **Sauts inconditionnels**

Cette instruction permet un branchement à une adresse donnée et ceci inconditionnellement. Le saut est donc toujours effectué.

JMP champ2	champ2 est une adresse les champs reg et m sont sans signification	saut inconditionnel à l'adresse champ2

JMP $(12CF)_{16}$: saut à l'adresse $(000012CF)_{16}$.

- **Sauts conditionnels**

Ces instructions permettent d'effectuer un branchement à une adresse donnée si une condition est réalisée. Ces conditions sont relatives aux indicateurs du registre d'état.
Si la condition n'est pas réalisée, l'exécution du code se poursuit en séquence.

JMPP champ2	Saut si positif champ2 est une adresse les champs reg et m sont sans signification	saut à l'adresse Champ2 conditionné au positionnement à 0 du bit S du registre PSW
JMPN champ2	Saut si négatif champ2 est une adresse les champs reg et m sont sans signification	saut à l'adresse Champ2 conditionné au positionnement à 1 du bit S du registre PSW
JMPO champ2	Saut si overflow champ2 est une adresse les champs reg et m sont sans signification	saut à l'adresse Champ2 conditionné au positionnement à 1 du bit O du registre PSW
JMPC champ2	Saut si carry champ2 est une adresse les champs reg et m sont sans signification	saut à l'adresse Champ2 conditionné au positionnement à 1 du bit C du registre PSW
JMPZ champ2	Saut si zéro champ2 est une adresse les champs reg et m sont sans signification	saut à l'adresse Champ2 conditionné au positionnement à 1 du bit Z du registre PSW

Les instructions de manipulation de la pile

Ces instructions permettent d'enregistrer un élément dans la pile ou d'ôter un élément de la pile. La pile est une zone mémoire gérée selon un ordre LIFO (Last In First Out).
Ces instructions ne modifient pas le registre d'état, PSW.

- **Enregistrer un élément dans la pile**

PUSH Rg1 reg	le contenu de reg est mis dans la pile

- **Oter un élément de la pile**

POP Rg1 reg	le sommet de la pile est rangé dans reg

Exercices

Exercice 1. Manipulation des modes d'adressages

A l'issue de l'exécution du code assembleur suivant et compte tenu de l'état initial de la mémoire et des registres du processeur, la case mémoire d'adresse 1000 a pour contenu la valeur

100 ? 'a'? 1998 ?

La représentation des nombres signés utilise la convention du complément à 2.

Adresse	Contenu
400	2000
404	412
408	d
412	3000
416	305

Registre	Contenu
RB	100

LOAD D R0 400
LOAD Im R1 1002
ADD Rg2 R0 R1
NEG Rg1 R1
ADD I R1 404
STORE B R1 900

Exercice 2. Manipulation des modes d'adressages

Les mots mémoire à partir de l'adresse 1000 contiennent des valeurs suivantes :

Adresse case mémoire	Contenu
1000	15
1004	45
1008	32
1012	1256
1016	253
1020	256
1024	996
1028	1024
1032	- 1
1036	996

On exécute le programme suivant, avec un format d'instruction et des modes d'adressage qui sont ceux de la machine MAP32 (machine du cours et des eds). Le registre RB est le registre de base
Décrivez ce que réalise chaque instruction du programme et dites globalement, ce que réalise ce programme.

```
            LOAD Im R1 6
            LOAD Im R0 0
            LOAD I RB 1028
Boucle:     LOAD B R2 4
            ADD Rg2 R0 R2
            ADD Im RB 4
            ADD D R1 1032
            JMPZ  Fin
            JMP Boucle
Fin         STORE B R0 16
```

Exercice 3

On vous demande de répondre à chacune de ces questions en recopiant sur votre copie la bonne réponse (soit VRAI, FAUX ou l'un des choix qui vous est donné). Si la réponse est fausse, énoncez ce qui est vrai.

Q1/ La lecture en mémoire centrale d'un opérande désigné par un mode d'adressage indirect nécessite :

a/ un accès mémoire b/ aucun accès mémoire c/ deux accès mémoire

Q2/ Le registre compteur ordinal contient la prochaine instruction à exécuter.

VRAI FAUX

Q3/ On considère une mémoire adressable par octets, composée de mots de 32 bits, d'une capacité égale à 4Mmots. La taille d'une adresse permettant d'adresser tous les octets de cette mémoire est :

a/ 20 bits b/ au minimum 24 bits c/ 22 bits

Exercice 4. Code en langage d'assemblage

Question 1

Ecrivez un programme en langage d'assemblage qui réalise le calcul suivant :

B = (A * 5) + (6 + B).

A et B sont deux variables correspondant chacune à un mot mémoire.

Question 2

Expliquez ce que fait le programme assembleur qui vous est donné sachant que la machine est sur 32 bits et travaille en complément à 2 pour la représentation des nombres signés. Les chiffres sont exprimés en base 10. La case mémoire d'adresse 1000 contient la valeur 2147483640 (2^{31} – 8)

Programme 1
0
4 LOAD D R1 1000
8 ADD Im R1 1
12 JMPO 20
16 JMP 8
20 STORE I R1 24
24 0

Exercice 5.

On considère une mémoire centrale chargée avec le programme suivant dont les instructions sont données en langage d'assemblage, selon le format établi pour MAP32.

Adresse mémoire	Mot mémoire	Commentaire éventuel
A :		Valeur de la case A non initialisée
B :		Valeur de la case B non initialisée
	IN D A	L'instruction IN permet la lecture au clavier d'une valeur pour A
	LOAD Im R1 -1	
	LOAD D R2 A	
	JMP Addition	
Fin	STORE D R2 A	
	STOP	Fin de l'exécution
Addition	ADD Rg2 R2 R1	
	JMP Fin	

Question 1

Complétez la colonne commentaire pour expliquer ce que réalise chaque instruction, puis concluez en expliquant ce que fait ce programme.

Question 2

On suppose que les nombres signés sont représentés en complément à 2 sur 8 bits .

A/ Le registre d'état PSW contient un ensemble d'indicateurs S, C, O, Z. Rappelez leur rôle.

B/ L'instruction IN D A, lit la valeur 1 au clavier. Expliquez quelle est la valeur contenue dans le registre PSW à la suite de l'exécution du programme. On prendra comme convention de valeur pour les indicateurs O, C, S, Z celle adoptée dans le cours.

C/ L'instruction IN D A, lit la valeur -128 au clavier. Expliquez quelle est la valeur contenue dans le registre PSW à la suite de l'exécution du programme. On prendra comme convention de valeur pour les indicateurs O, C, S, Z celle adoptée dans le cours.

Question 3

Lorsque l'opération d'addition produit un overflow, on souhaite écrire le résultat contenu dans R2 à l'adresse B. Modifiez le code du programme pour permettre cette opération.