Chapitre 5: La machine MIASM

- Introduction .
- Architecture générale de MIASM.
- Format d'une instruction et modes d'adressage de MIASM
- Jeu d'instructions de MIASM .
- Programmation en langage MIASM .

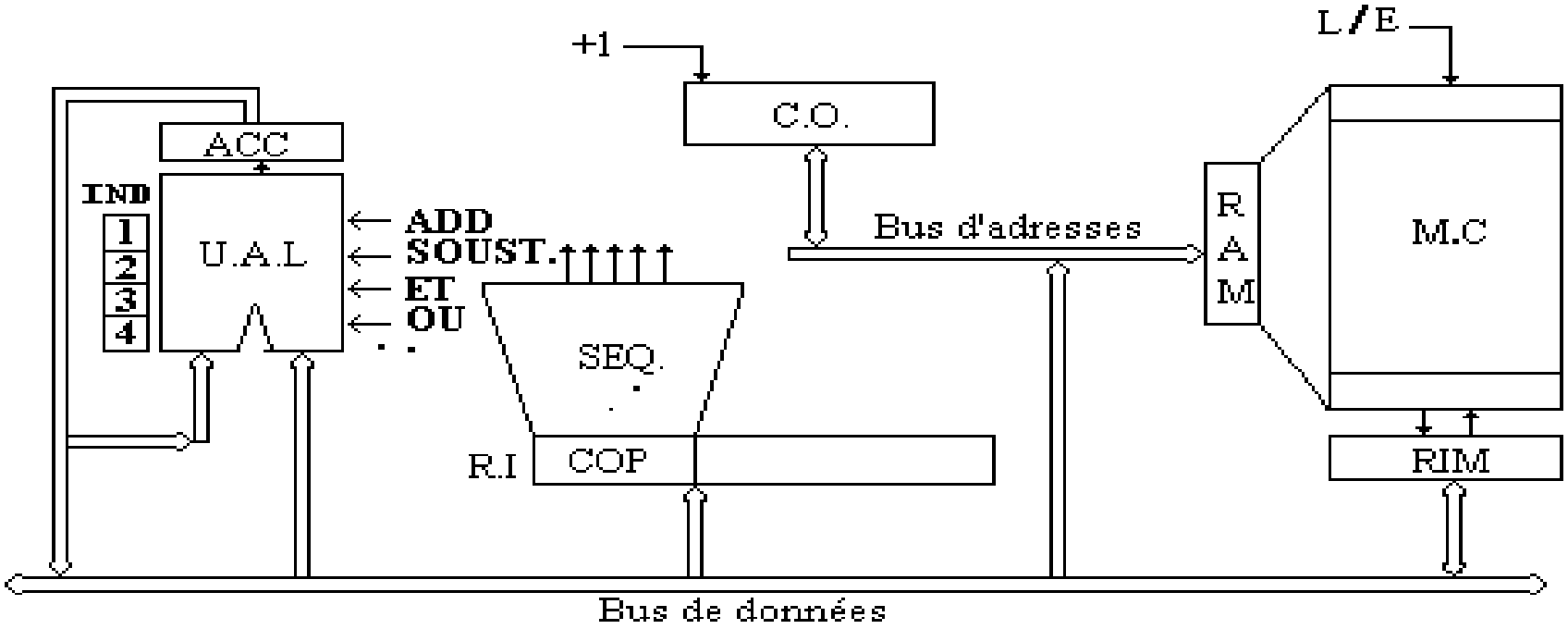
Introduction

- Dans ce chapitre, on va montrer le fonctionnement complet d'un ordinateur à travers une machine fictive (donc pas réelle), «pédagogique », que nous appellerons « MIASM » pour **M**achine d'**I**nitiation **A** la **S**tructure **M**achine.

- Cette machine est très simplifiée, et n'est, donc pas une machine réelle (non implémentée).

- Cependant, elle possède tous les composants et toutes les caractéristiques d'un véritable ordinateur.

1. Structure générale de MIASM



La taille de la mémoire est de 2048 mots → bus d'adresse sur 11 bits
La taille d'un mot est de 16 bits → bus de données sur 16 bits
La taille de l'accumulateur , le RIM et le RI est de 16 bits
La taille du CO et de RAM est de 11 bits
La machine possède 4 indicateurs (flags)

Les indicateurs

- L‘indicateur **N° 1**: est mis à **1** si un **débordement** de capacité se produit dans une opération , il est mis à **0** dans le cas normal.

- L'indicateur **N° 2**: est mis à **1** si l’opération dégage **une retenue**, à **0** sinon.

- L'indicateur **N° 3**: est mis à **1** si le contenu de l‘accumulateur est **égal à zéro.** Il est mis à **0** si le contenu de l'accu est non nul.

- L'indicateur **N° 4**: est mis à **1** si le contenu de l'accu **est Négatif**, il

  est à **0** sinon.

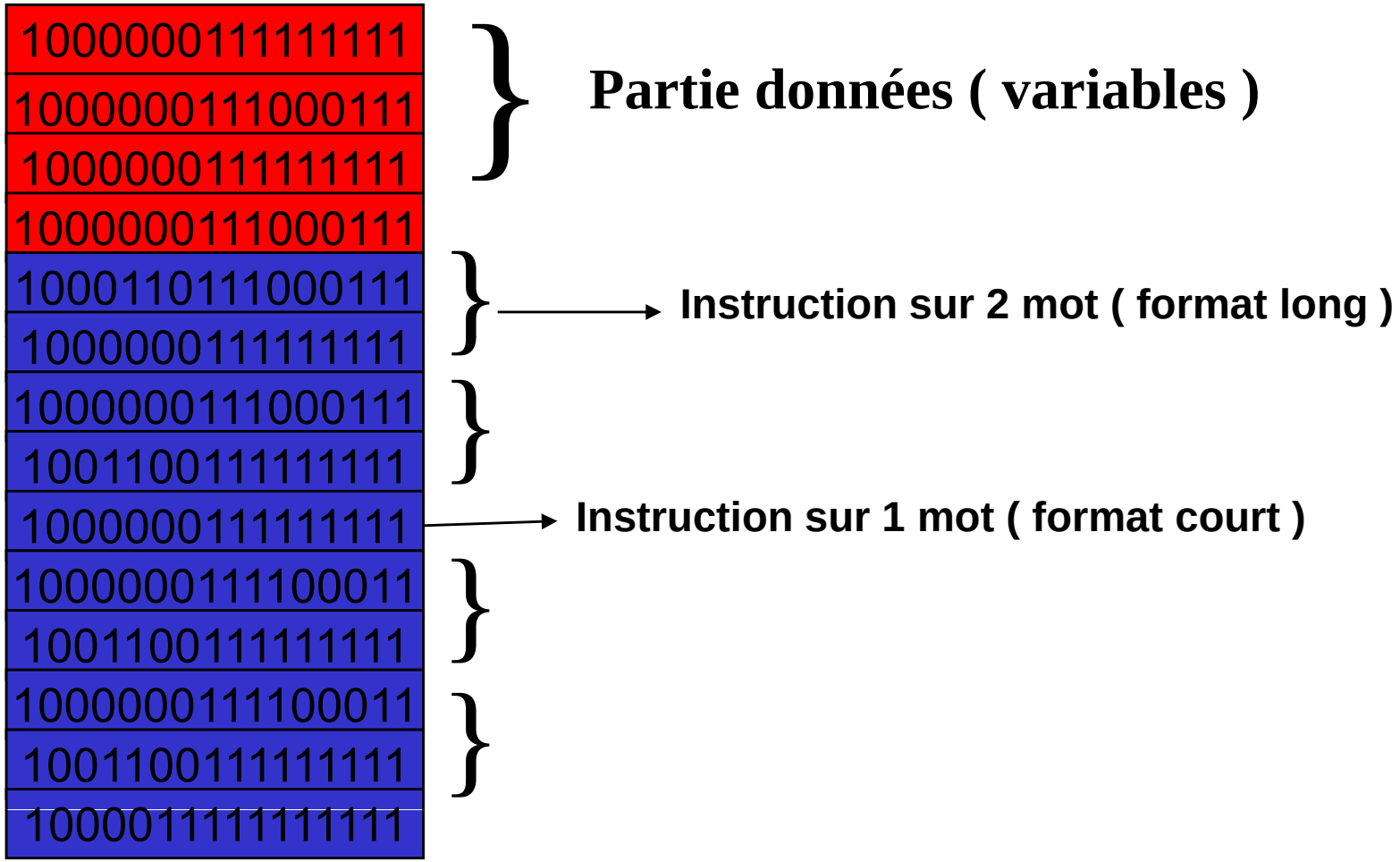
2. Format d'une instruction

- **MIASM** est une machine à une adresse.
- Ses instructions sont représentées en binaire sur un ou plusieurs mots.
- Elle dispose de deux types de format d'instructions:

  – Les instructions dites de format LONG, occupent deux mots mémoire :
    - Le premier mot comporte le code opération, le type d'adressage, …
    - le deuxième mot comporte l'adresse de l'opérande.

  – Les instructions dites de format COURT: Elles occupent un seul mot (comporte le code opération, le type d'adressage,…….).

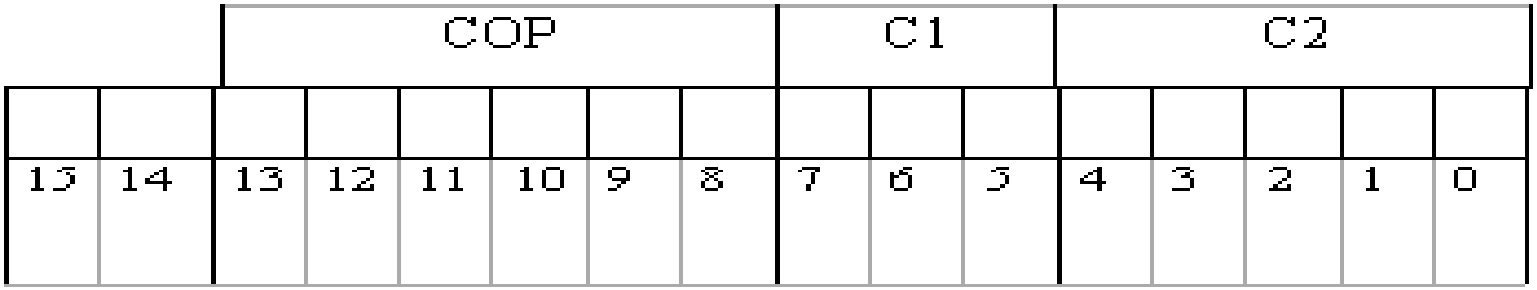
    Ce format est utilisé par les instructions qui ne comportant pas une partie adresse (Exemple : entrées/sorties).



Format d'une instruction

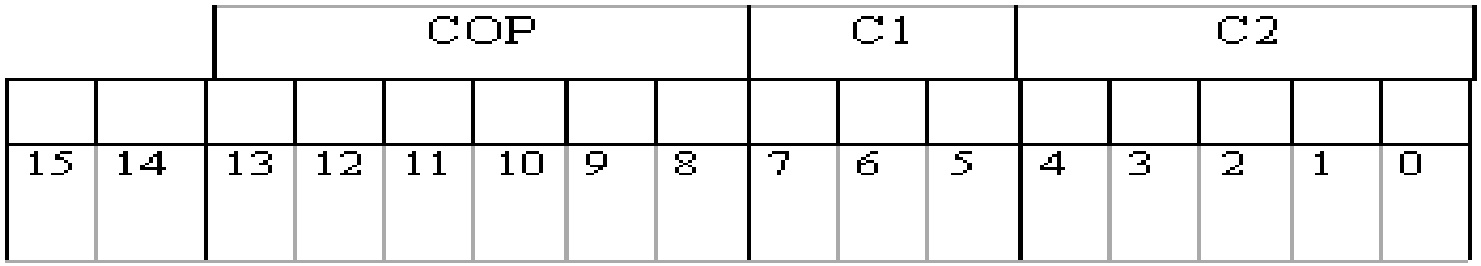
2.1 Le premier mot

- Ce mot est commun aux deux types d'instructions.
- Il tient sur 16 bits et il est divisé en plusieurs champs.



Les bits 15 et 14: Servent à indiquer le type d'adressage:

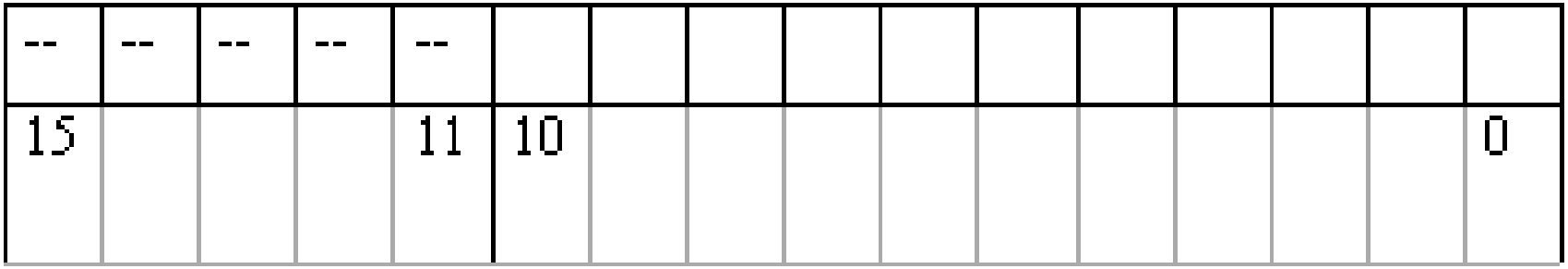
00 : adressage direct .
01 : adressage indirect.
10 : adressage immédiat;
11 : Configuration interdite.



- Les bits 13 à 8: donnent (sur 6 bits) Le CODE OPERATION à effectuer.
- Le bit 13, premier bit du code opération, indique le format de l'instruction :
  - bit 13=0 ==> instruction format court.
  - bit 13=1 ==> instruction format long.
- Les bits 7 à 5: Définissent une zone ou champ appelé C1 qui désigne le N° de l'indicateur à tester pour les instructions conditionnelles.
- Les bits 4 à 0: Définissent une zone ou champ appelé C2 qui désigne le N° du périphérique pour les instructions d'Entrée/Sortie.

Format d'une instruction
2.2. Le deuxième mot

- Propre aux instructions de format long, il contient:
  - La partie adresse de l'instruction qui tient sur 11 bits (bit 0 au bit 10), et désigne l'emplacement en mémoire où se trouve l'opérande.
  - L'opérande sur 16 bits (bit 0 au bit 15), cas d'instruction immédiate.



Quelques conventions

Pour décrire l'effet des instructions, nous adopterons les conventions suivantes:

- ADR : désigne une adresse en mémoire ;
- (ADR) : désigne le contenu de la cellule mémoire d'adresse ADR,
- R : désigne une adresse de registre ;
- (R) : désigne le contenu du registre d'adresse R, exemple: ACC, CO, …
- Signe ← placé entre deux membres indique que l'information définie dans le deuxième membre est rangée dans l'élément de mémoire (ADR ou R) défini par le premier membre;
- AE: Adresse Effective désigne la dernière adresse contenant l'opérande;

- **Exemple1** : Déroulement de l'instruction d'addition en mode immédiat sur MIASM dont l'effet est: ACC←(ACC)+ Valeur

  - **Phase 1** : (rechercher l'instruction à traiter)
    - Mettre le contenu du CO dans le registre RAM : *RAM←(CO)*
    - Commande de lecture à partir de la mémoire : *Lect*
    - Transfert du contenu du RIM dans le registre RI : *RI←(RIM)*
    - Analyse et décodage
  - **Phase 2** : (rechercher opérande et traitement)
    - *CO ←(CO)+ 1*
    - Transfert de l 'adresse du 2eme mot dans le registre RAM : *RAM←(CO)*
    - Commande de lecture à partir de la mémoire : *Lect*
    - Transfert de l'opérande vers l'UAL : *UAL ←(RIM)*
    - Commande de l'exécution de l'opération : (*add*)
  - **Phase 3** : (passer à l'instruction suivante)
    - *CO ←(CO)+ 1*

- **Exemple 2 :** déroulement de l'instruction d'addition en mode direct sur MIASM

  Effet: ACC←(ACC)+ (ADR)

  - **Phase 1 :** (rechercher l'instruction à traiter)
    - Mettre le contenu du CO dans le registre RAM : *RAM←(CO)*
    - Commande de lecture à partir de la mémoire *: Lect*
    - Transfert du contenu du RIM dans le registre RI : *RI←(RIM)*
    - Analyse et décodage
  - **Phase 2 :** (rechercher opérande et traitement)
    - *CO ←(CO)+ 1*
    - Transfert de l 'adresse du 2eme mot dans le registre RAM : *RAM ←(CO)*
    - Commande de lecture à partir de la mémoire : *Lect*
    - Transfert de l'adresse de l'opérande vers le RAM : *RAM ←(RIM)*
    - Commande de lecture à partir de la mémoire : *Lect*
    - Transfert du contenu du RIM (l'opérande)vers l'UAL : *UAL ←(RIM)*
    - Commande de l'exécution de l'opération : *(add)*
  - **Phase 3 :** (passer à l'instruction suivante)
    - *CO ←(CO)+ 1*

- **Exemple3** : déroulement de l'instruction au format court sur MIASM (exemple entrées /sorties).

  - **Phase 1** : (rechercher l'instruction à traiter)
    - Mettre le contenu du CO dans le registre RAM : *RAM ← (CO)*
    - Commande de lecture à partir de la mémoire : *Lect*
    - Transfert du contenu du RIM dans le registre RI : *RI ← (RIM)*
    - Analyse et décodage
  - **Phase 2** : (traitement)
    - Commande de l'exécution de l'opération
  - **Phase 3** : (passer à l'instruction suivante)
    - *CO ← (CO)+ 1*

3. Le jeu d'instructions de MIASM

- Pour pouvoir exécuter des programmes sur la machine MIASM, on dispose d'un certain nombre d'instructions qui forment le langage de la machine.
- Ce langage est un langage ASSEMBLEUR.

```
          ORG   X'100'
X         RM    1
Y         RC    10
DEBUT     ENT    01
          ADM    Y
          RGM    X
          SOR    02
          STOP
          END   DEBUT
```

Langage assembleur


```
1110000011101011
0010110011101011
0010000011101011
1110000011101011
1100000011101011
1000000011101011
1110000011101011
```

Langage machine

Syntaxe de l'instruction MIASM

Une instruction MIASM est écrite sous la forme suivante:

[**étiquette**] **Mnémonique** [, **n°**] [*****] **Argument** /* Commentaire */

- *Etiquette* : Symbole désignant une instruction ou donnée: facultatif;
- *Mnémonique:* Symbole associé au code opération.
  Pour les instructions de branchement conditionnel, le **n°** (facultatif) représente l'indicateur à tester;
- *Argument:* Désigne une adresse (directe ou indirecte, si précédée par *****) ou une donnée dans le cas d'opération immédiate;
- *Commentaire:* Texte ajouté pour précision (facultatif)

3.1 Les instructions d'échange entre l'accumulateur et la mémoire centrale

- **Instruction** : **RANGEMENT (RGM)**
- *Effet* : Le contenu de l'accumulateur est écrit en mémoire centrale à l'adresse figurant dans l'instruction. Le contenu de l'accumulateur n'est pas modifié. AE← (ACC)
- *Format* : Long
- *Adressage* : Direct ou Indirect

- **Exemple** :
  - RGM A (mode direct)
  - RGM *B (mode indirect)

- **Instruction** : **CHARGEMENT IMMEDIAT (CHI)**
- *Effet* : La partie adresse de l'instruction est chargée dans l'accumulateur en remplacement du précédant:

  ACC← Opérande
- *Format* : Long
- *Adressage* : Immédiat
- Les indicateurs 3 et 4 de l'UAL sont positionnés selon l'information chargée.

- **Exemple** :
  - CHI 12
  - CHI 0

- **Instruction** : **CHARGEMENT MOT (CHM)**
- *Effet* : Le contenu du mot mémoire référencé par la partie adresse de l'instruction est chargé dans l'accumulateur:

  ACC ← (AE)
- *Format* : Long
- *Adressage* : Direct ou Indirect
- Les indicateurs 3 et 4 de l'UAL sont positionnés selon l'information chargée.
- **Exemple** :
  - CHM A (mode direct)
  - CHM *B (mode indirect)

3.2 Instructions d'opérations arithmétiques

- **Instruction** : **ADDITION / SOUSTRACTION IMMEDIATE (ADI / SI)**
- *Effet* : La partie adresse de l'instruction est additionnée/soustraite au/du contenu de l'accumulateur. Le résultat est dans l'accumulateur: ACC← (ACC) +/- Opérande
- *Format* : Long
- *Adressage* : Immédiat
- *Observations* : Les indicateurs 1,2,3 et 4 de l'UAL sont positionnés selon l'information chargée.

- **Exemple** :

  ADI 12

  SI 13

- **Instruction** : **ADDITION / SOUSTRACTION MOT (ADM / SM)**

- *Effet* : Le contenu du mot mémoire référencé par la partie adresse de l'instruction est additionné/soustrait au/du contenu de l'accumulateur. Le résultat est dans l'accumulateur:

  ACC← (ACC) +/- (AE)

- *Format* : Long
- *Adressage* : Direct ou Indirect
- *Observations* : Les indicateurs 1,2,3 et 4 de l'UAL sont positionnés selon l'information chargée.
- **Exemple :**

  ADM A

  SM *B

3.3 Instructions d'opérations logiques

- **Instruction** : **ET MOT**

- *Effet* : Un ET logique est effectué entre le contenu de l'accumulateur et le contenu du mot adressé par la partie adresse de l'instruction. Le résultat est dans l'accumulateur.

  ACC← (ACC) et (AE)

- *Format* : Long
- *Adressage* : Direct ou Indirect
- *Observations* : Les indicateurs 3 et 4 de l'UAL sont positionnés selon le résultat trouvé.
- **Exemple** :

  ET A

  ET *B

- **Instruction** : **OU / OUX MOT**

- *Effet* : Un OU / OUX logique est effectué entre le contenu de l'accumulateur et le contenu du mot adressé par la partie adresse de l'instruction. Le résultat est dans l'accumulateur.

ACC← (ACC) ou/oux (AE)

- *Format* : Long
- *Adressage* : Direct ou Indirect
- *Observations* : Les indicateurs 3 et 4 de l'UAL sont positionnés selon le résultat trouvé.
- **Exemple** :

```
OU   A
OU  *B
```

- **Instruction** : **NON MOT**

- *Effet* : Tous les bits du contenu du mot adressé sont inversés.

  ACC← non (AE)

- *Format* : Long
- *Adressage* : Direct ou Indirect
- *Observations* : Les indicateurs 3 et 4 de l'UAL sont positionnés selon le résultat trouvé.
- **Exemple** :

  NON A

  NON *B

3.4 Instruction de rupture de séquence

- Egalement appelée instruction de branchement ou de saut;

- Permet de modifier le déroulement séquentiel du programme, en faisant suivre l'instruction par celle dont l'adresse est fournie par le deuxième mot de l'instruction de rupture de séquence;

- Le branchement peut être conditionnel ; il ne sera alors effectif que si une condition (exprimée par le N° d'indicateur), portant sur le contenu de l'accumulateur, est réalisée;
  Sinon le programme continuera en séquence;

- **Si** la réponse de l'UAL est que la condition est réalisée,

  Alors, l'unité de contrôle valide le transfert de l'adresse vers le CO, et inhibe l'addition de 1 au CO;

  Sinon, elle commande uniquement l'incrémentation de 1 du CO.

Instructions de branchement

- **Instruction** : BRANCHEMENT SI CONDITION VERIFIEE **(BCV,ind)**

- *Effet* : Les trois bits du champ C1 donnent le N° de condition de 0 à 4 à tester :
  - Si le N° de la condition est 0 (branchement inconditionnel): exécution d'un branchement à l'adresse effective AE: CO ← AE
  - Si le N° de la condition est i =1.2.3 ou 4 : alors tester l'indicateur correspondant et exécution d'un branchement à l'adresse effective AE si l'indicateur est à 1; CO ← AE
  - Si l'indicateur est à 0, poursuivre en séquence (non branchement).

CO ← (CO) + 1

- **Exemple**

BCV,4 branchement si l'indicateur 4 est à 1 (le résultat est négative)

BCV,3 branchement si indicateur 3 est à 1 (le résultat est nul)

- **Instruction** : BRANCHEMENT SI CONDITION FAUSSE (BCF,ind)

- Effet : Les trois bits du champ C1 donnent le N° de condition de 0 à 4 à tester :
  - Si le N° de la condition est 0 (branchement inconditionnel) : exécution d'un branchement à l'adresse effective AE: CO ← AE
  - Si le N° de la condition est 1.2.3 ou 4 : tester l'indicateur correspondant et exécution d'un branchement à l'adresse effective AE si l'indicateur est à 0: CO ← AE
  - Si l'indicateur est à 1 poursuivre en séquence (non branchement).

    CO ← (CO) + 1

- **Exemple**

  BCF,4 branchement si l'indicateur 4 est à 0 (le résultat n'est pas négative)

  BCF,3 branchement si indicateur 3 est à 0 (le résultat n'est pas nul)

- **Exemple :** déroulement de l'instruction de branchement si la condition est vérifiée (exemple tester l'indicateur 4 s'il est égale à 1, alors se brancher):

BCV,4 ADR

- **Phase 1 :** (rechercher l'instruction à traiter)
  - Mettre le contenu du CO dans le registre RAM : *RAM ← (CO)*
  - Commande de lecture à partir de la mémoire : *Lect*
  - Transfert du contenu du RIM dans le registre RI *: RI ← (RIM)*
  - Analyse et décodage
  - *CO ← (CO)+ 1*

Si condition vérifiée (valeur de l'indicateur 4 est égale à 1)

- **Phase 2 :** (traitement)
  - Transfert de l 'adresse du 2eme mot dans **le registre RAM** : *RAM ← (CO)*
  - Commande de lecture à partir de la mémoire : *Lect*
  - Transfert de l'adresse de l'instruction vers le **CO** : *CO ← (RIM)*

Si condition non vérifiée

- **Phase 3 :** (passer à l'instruction suivante)
  - *CO ← (CO)+ 1*

3.5 Instructions d'entrées/sorties

- **Instruction** : **ENTREE DE DONNEES ENT**

- *Effet* : Une donnée est entrée à partir d'un périphérique dans l'accumulateur: ACC ← information lue
- *Format* : Court
- *Adressage* : Immédiat
- *Observations* : - Le champ C1 n'est pas utilisé;
  - Le champ C2 donne le numéro du périphérique
  (le périphérique 01 indique le clavier)

Exemple :

```
ENT   01
RGM   A
```

- **Instruction** : **SORTIE DE DONNEES SOR**

- *Effet* : Une donnée est sortie de l'accumulateur vers un périphérique: information à sortir ← (ACC)
- *Format* : Court
- *Adressage* : Immédiat.
- Observations : - Le champ C1 n'est pas utilisé
  - Le champ C2 donne le numéro du périphérique **(le périphérique 02 indique l'écran)**

- **Exemple**

```
CHM  A
SOR  02
```

3.6 Instructions d'arrêt du programme.

- **Instruction** : **FIN DU PROGRAMME (STOP)**

- *Effet* : Provoque un arrêt du programme en cours d'exécution.
- *Format* : Court
- *Adressage* : Immédiat

Instruction	Mnémonique	Cod. Op.	Format	Adr
Rangement mot	RGM	20	L	D/I
Chargement mot	CHM	22	L	D/I
Chargement immédiat	CHI	21	L	Imm
Addition immédiate	ADI	23	L	Imm
Addition mot	ADM	24	L	D/I
Soustraction immédiate	SI	25	L	Imm
Soustraction mot	SM	26	L	D/I
ET mot	ET	27	L	D/I
OU mot	OU	28	L	D/I
OU Exclusif mot	OUX	29	L	D/I
NON mot	NON	2A	L	D/I
Branchement si condition vraie	BCV	2B	L	D/I
Branchement si cond fausse	BCF	2C	L	D/I
Entrée	ENT	01	C	--
Sortie	SOR	02	C	--
Stop	STOP	00	C	--

L : Long, C : Cours;

D : Direct; I : Indirect; Imm. : Immédiat.

4. Structure générale d'un programme en langage MIASM

- Un programme écrit afin d'être exécuté sur MIASM est composé de deux parties :
  - partie données
  - et partie instructions

- **Exemple :**

```
        ORG   X'100'
X       RM    1
Y       RC    10
-------------------------
DEBUT   ENT    01
        ADM    Y
        RGM    X
        SOR    02
        STOP
-------------------------
        END    DEBUT
```

•Adresse début du programme en mémoire

•Réservation d'un mot mémoire

•Réserver un mot mémoire et l'initialiser avec la valeur 10

Les instructions

4.1 Partie données

- Pour les données on utilise les directives:
  - RM permet de réserver une zone mémoire de N mots mémoire.
  - RC permet de réserver une zone mémoire avec initialisation.
  - ORG permet d'implanter le programme à partir de l'adresse définie

- **Exemple**

```
    ORG  100
X   RM    1                    réserver un seul  mot
Y   RC    23                   réserver un mot et l'initialiser par la valeur 23
Z   RM    4                    réserver 4 mots mémoire
T   RC    X'AB'  X'10'  X'23'  réserver 3 mots mémoire initialisés avec les valeurs
                                 hexadécimales 'AB' , '10' et '23'
```

4.2 Partie instructions

- La partie instruction contient l'ensemble des instructions (dans l'ordre) qui détermine la logique du programme.
- Dans cette partie on peut trouver des instructions:
  - arithmétiques ,
  - logiques ,
  - d'entrées /sorties ,
  - d'échange,
  - De rupture de séquence.

Exemple 1

- Soit l'algorithme suivant :

```
                ORG    100
A               RM     1     /* Réserver A*/
B               RM     1     /* Réserver B*/
C               RM     1     /* Réserver C*/
```

```
DEBUT           ENT    01
                RGM    B
                ENT    01
                RGM    C
                CHM    B
                ADM    C
                SI     123
                RGM    A
                SOR    02
                STOP
                END    DEBUT
```

Lire (B)

Lire (C)

A← (B+C) -123

Écrire (A)

Exercice :

Quel serait le contenu du Mot " RESU " à la fin de l'exécution du programme suivant:

```
        ORG   0
DON     RC    X'ABCD'
RESU    RM    1
DEBUT   CHI   X'F00F'
        ET    DON
        ADI   X'2FFD'
        RGM   RESU
        CHI   X'0FF0'
        ET    DON
        ADI   X'00F0'
        OU    RESU
        RGM   RESU
        END   DEBUT
```

Exemple d'utilisation de BCV

En langage de transfert:

Si A >= B alors Z← A+B;

A← A+1;

Si (A - B) >= 0 alors Z← A+B;

A← A+1;

En langage MIASM :

```
        CHM   A
        SM    B      /* A-B */
        BCV,4 suite
        CHM   A
        ADM   B
        RGM   Z
Suite   CHM   A
        ADI   1
        RGM   A
```

La forme SI …… SINON

- **Si cond alors** Action 1
  sinon Action 2

- Évaluation de la **condition**
  - **Si** condition est fausse branchement aller à A2
  - Exécuter Action 1
  - Branchement inconditionnelle à suite

A2 : Exécuter Action 2

Suite: …

Exemple d'utilisation de Si

En langage de transfert

Si A > B **alors** Max ← A
sinon Max ← B

Si (A – B) > 0 **alors** Max ← A
sinon Max ← B

En MIASM

```
        CHM     A
        SM      B
        BCV,4   A2
        CHM     A
        RGM     MAX
        BCV,0   suite
A2      CHM     B
        RGM     MAX
Suite   CHM     MAX
        SOR     02
```

Condition composée

- **Si cond1 et cond2 alors** Action
  sinon aller à Suite

  Évaluation de la **condition 1**
  Si condition 1 est fausse branchement à suite
  Evaluation de la **condition 2**
  Si condition 2 est fausse branchement à suite
  Exécuter Action
  Suite: …

- Si **cond1 ou cond2 alors** Action
  sinon aller à Suite

  Évaluation de la **condition 1**
  Si condition 1 est vraie branchement à Action
  Evaluation de la **condition 2**
  Si condition 2 est fausse branchement à suite
  Action : Exécuter Action
  Suite: …

Exemple 1: Condition composée

En langage de transfert

Si (A >= B) et (A < C) **alors**
RES←C

En MIASM

```
        CHM     A
        SM      B
        BCV,4   suite
        CHM     A
        SM      C
        BCF,4   suite
        CHM     C
        RGM     RES
suite   …
```

Exemple 2: Condition composée

En langage de transfert

Si A >= B ou A < C **alors**
Res← B+C

En MIASM

```
        CHM     A
        SM      B
        BCF,4   action
        CHM     A
        SM C
        BCF,4   suite
Action  CHM     B
        ADM     C
        RGM     RES
Suite ……
```

Forme Tant que

Exemple : Soit

res ← 1+2+3+4+5+6+7+8+9

K←1

RES← 0

Tant que K< 10 **faire** /* (K-10 < 0)

Début

RES← RES + K;

K←K+1;

End

```
        Org     X'100'
K       RM      1
RES     RM      1
Debut   CHI     0
        RGM     RES
        CHI     1
        RGM     K
BOUCLE  SI      10
        BCF,4   FIN
        CHM     RES
        ADM     K
        RGM     RES
        CHM     K
        ADI     1
        RGM     K
        BCV,0   BOUCLE
FIN     CHM     RES
        SOR     02
        STOP
        END     Debut
```

Forme répéter

Exemple

res ← 1+2+3+4+5+6+7+8+9

K←1

RES← 0

Répéter

RES← RES + K;

K←K+1;

Jusqu'à K=10

```
        Org     X'100'
K       RM      1
RES     RM      1
Debut   CHI     0
        RGM     RES
        CHI     1
        RGM     K
BOUCLE  CHM     RES
        ADM     K
        RGM     RES
        CHM     K
        ADI     1
        RGM     K
        SI      10
        BCV,3   FIN
        BCV,0   BOUCLE
FIN     CHM     RES
        SOR     02
        STOP
        END     Debut
```