

ARCHITECTURE

TD3

Exo1 :

- 1- Pour réaliser $X = P + Q$

```
LDA P
ADD Q
STA X
```

- 2- Pour: si $X = 4$ alors $Y = 6$
Sinon $Y = Q + 2$

```
CMP #4
BRZ 5
LDA Q
ADD #2
STA Y
STOP ou BRA 3 //amène au stop ^^
SET #6
STA Y
STOP
```

- 3-

FETCH / DECODE / EXECUTE

Fetch = charger l'instruction de la mémoire vers le registre d'exécution

- PC $\rightarrow$ MAR (PC = pointeur sur l'instruction suivante)
- Déclenchement d'1 lecture mémoire
- @ vient de MAR
Résultat de la lecture $\rightarrow$ MBR
- BR $\rightarrow$ UAL (Unité algorithmique et logique)
- Opération de l'UAL MBR ou 0 (zéro)
- UAL $\rightarrow$ IR
- PC $\leftarrow$ PC + 1

(d, e et f peuvent se résumer à IR $\leftarrow$ MBR)

Oui, et en plus c'est toujours le même chemin ^^

- 4- LDA X : phase EXECUTE

```
MAR  $\leftarrow$  IR<opérande>
MBR  $\leftarrow$  M<MAR>
D0  $\leftarrow$  MBR
```

STA X : phase EXECUTE

```
MBR  $\leftarrow$  D0
MAR  $\leftarrow$  IR<opérande>
M<MAR>  $\leftarrow$  MBR
```

ADD X : phase EXECUTE
 $MAR \leftarrow IR<oprande>$
 $MBR \leftarrow M<MAR>$
 $D0 \leftarrow UAL - D0 + MBR$

SET #N : phase EXECUTE
 $D0 \leftarrow IR<oprande>$ //adressage immédiat

ADD #N :
 $MBR \leftarrow Ir<oprande>$
 $dD0 \leftarrow UAL - D0 + MBR$

CMP #N:
 $MBR \leftarrow IR<op>$
 $CCR(Z) \leftarrow UAL - D0 - MBR$ //CCR cf sujet c'est expliqué // (mise à jour de Z)

BRZ
 Si $CCR(Z) = 1$
 $PC \leftarrow PC + IR<op>$

BRA
 $PC \leftarrow PC + IR<op>$

TD4 :

- 1) Machine à chargement/rangement : les seules opérations avec la mémoire sont les transferts (de mémoire à registre ou de registre à mémoire), tous les calculs sont effectués avec les registres et valeurs immédiates.
- 2) MOVE : op1 , op2

Op1 : @ , registre ou valeur immédiate
 Op2 : @ ou registre
 //@ à @ NON !! on passe par le registre^^

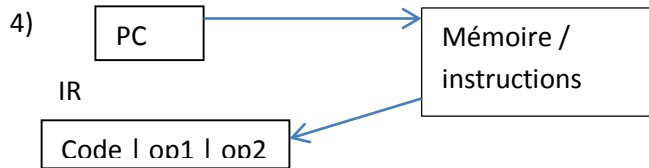
Pour les opérations arithmétique et logique (comme ADD) : op1, op2

Op1 : registre ou valeur immédiate // les calculs se font sur registre !!!
 Op2 : registre
 // c'est l'op2 qui stocke le résultat^^

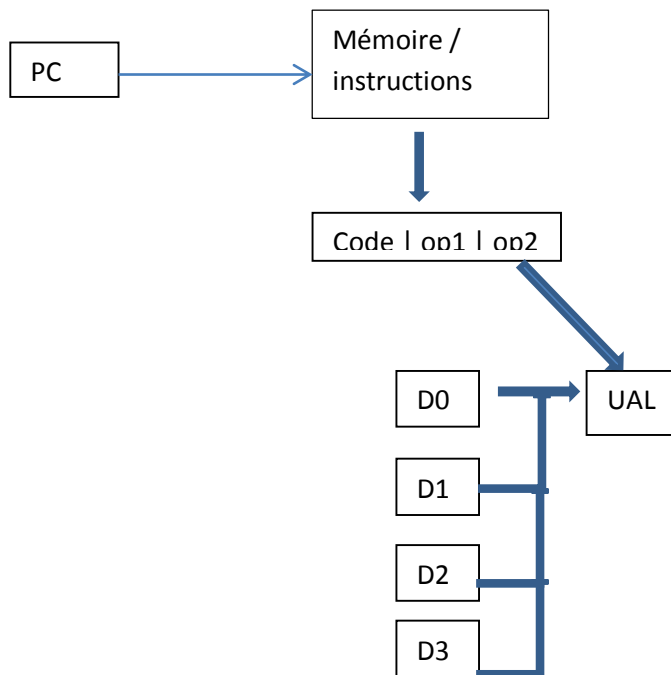
Pour le NOT : op1

Op1 : registre
 Pour BRA, BRZ, BRC, BRN : exemple : BBZ op
 Op : valeur immédiate
 Pour CMP : op1, op2
 Op1 : registre, valeur directe
 Op2 : registre

3) Pendant les phases : FETCH + EXECUTE



5) En entrée de l'UAL, on retrouvera : les registres de données et les valeurs immédiates, qui seront toujours données par la première opérande.

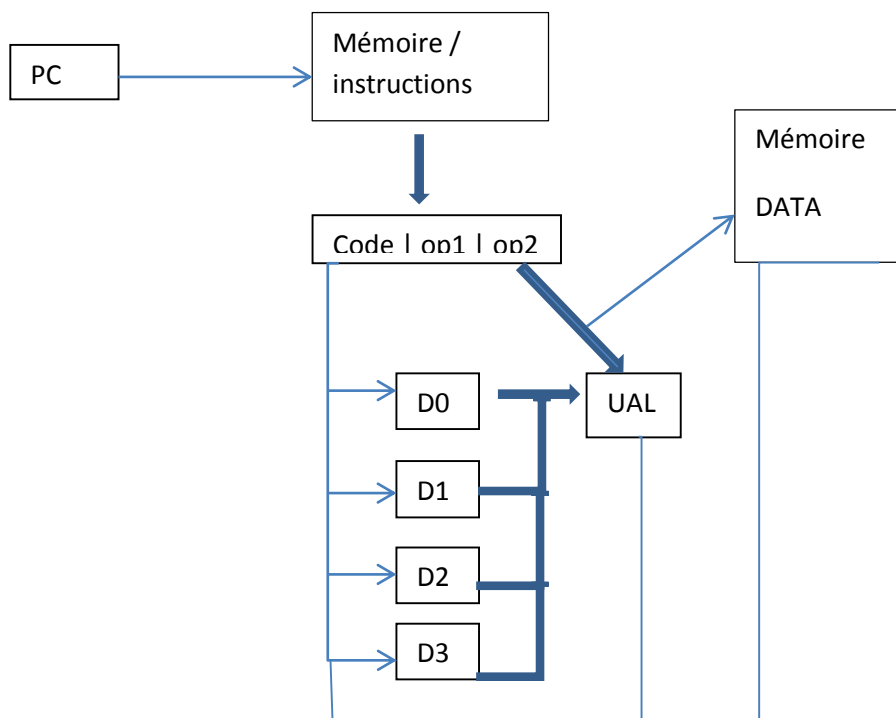


6) On peut y stocker : le résultat d'un calcul (sortie de l'UAL)

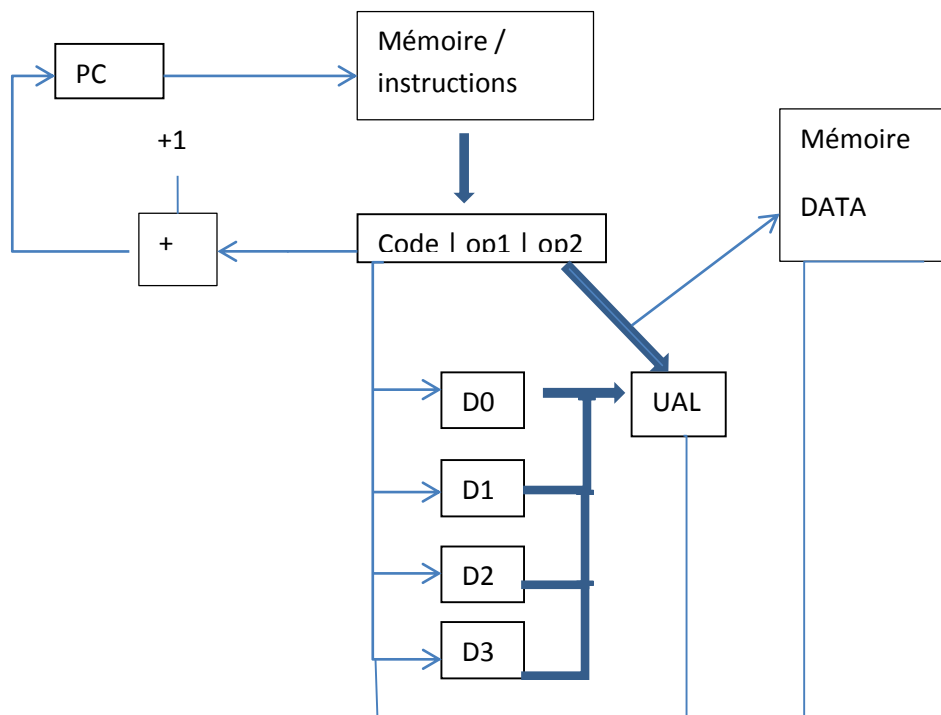
Une valeur provenant de la mémoire de données

Une valeur immédiate (op1) ou la valeur d'un registre (en passant par l'UAL par exemple, mais ce n'est pas obligé)

7) Les accès en mémoire ont lieu uniquement lors de transferts de registre à mémoire ou de mémoire à registre ou de valeur immédiate à mémoire



8)



9) Il y a 5 codes différents : v# - r, v# - @, @ - r, r - r, r - @

5 codes ≠ pour MOVE = 5

2 codes # par opération A et L = 14

1 code pour NOT = 1

2 codes pour CMP = 2

1 code par branchement = 4

= 26 en tout => 5 bits

TD 1 :

Exercice 1 : arithmétique et complément en 2

a) Le premier bit est le bit de signe

Les valeurs en complément à 2 vont de $2^{(n-1)} < x < 2^{(n-1)} - 1$

N= 5

b) $X_1 = 01100 = 12$

$X_2 = 01101 = 13$

$X_3 = 10111 = -9 = 2^{(n-1)} + 7$

$X_4 = 10100 = -12 = 2^{(n-1)} + 4$

c) $X_2 - X_1$

01101

-01100

=00001

$X_1 - X_2$

$x_2 \text{ C\`a}2 = 01101$

01100

-10011

=-10001 = -1

$X_1 + x_3$

01100

+10111

=00011 = 3

$x_1 + x_2$

01100

+01101

=11001 = -7

$X_3 + x_4$

10111

+10100

=01011 = 11

d) C'est nul il est overflow^^

e) $V_1 = 110 + 001 = 0$

$V_2 = 101 + 010 = 0$

$V_3 = 101 + 010 = 0$

$V_4 = 111 + 000 = 1$

$V_5 = 000 + 111 = 1$

Le V permet de voir le signe après calcul !!!

Exercice 2 :

Additionneur binaire avec retenue

Exercice 4 :

- a) 4 mots de 3bits donc $4 \times 3 = 12$ fils
- b) Le bus de données à besoin de 3 fils pour les mots mémoire de 3bits
 - i) 8 fils de données
10 fils d'@
 1024×8 bascules = 8192 bascules
8 OU à 1024 entrées
 $1024 + 1024 \times 8 + 2 = 9218$ ET