Chap. V Architecture

BRUNO DARID

13 août 2019

PLAN

1 Repères historiques

Les années 40 furent riches en termes de développement des premiers calculateurs programmables. Cependant, les programmes étaient externes (sur *cartes perforées*) avec tous les inconvénients que cela suppose (*vitesse d'exécution notamment*).
En 1945, un brillant mathématicien physicien, John von Neumann (fig. 1) décrit dans un rapport la structure d'un nouveau calculateur : le **calculateur à programme enregistré**. Il dirigea sa construction jusqu'en 1952.

2 Modèle d'architecture séquentielle (*dite de von Neumann*)

Dans son rapport de 1945, John von Neumann énumère les principaux organes de cette nouvelle machine :

— le processeur ;

— la mémoire ;

— les dispositifs d'entrée/sortie

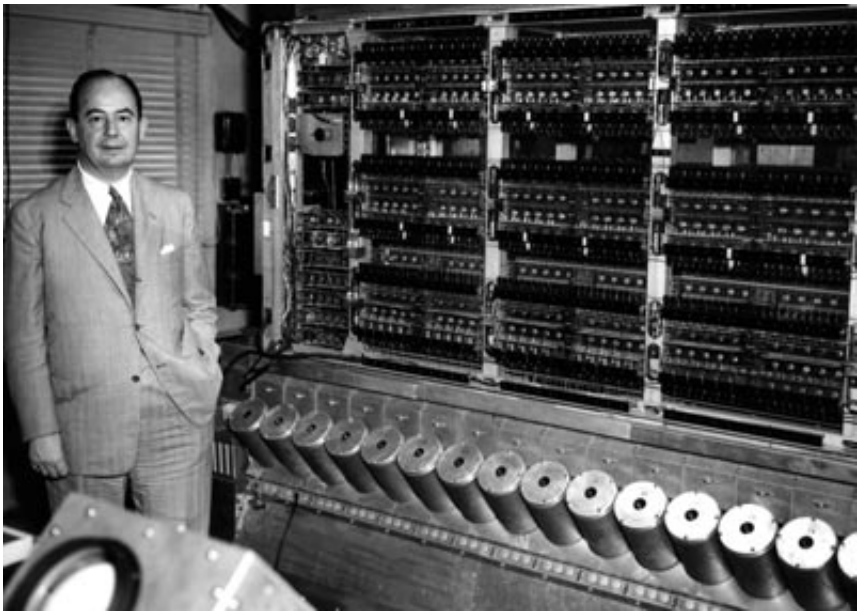

Fig. 1 – John Von Neumann

L'idée fondamentale est de stocker les données **et** les instructions des programmes en mémoire centrale.

2.1 Les composants essentiels

En juin 1945 dans la première version d'un rapport sur la conception de l'EDVAC John von Neumann décrit un schéma d'architecture d'un calculateur organisé autour des éléments suivants :

— une unité arithmétique et logique (*UAL*) ;

— une unité de commande (*Control Unit*) ;

— la mémoire ;

— des unités d'entrées/sorties.

Ces éléments étant reliés entre eux par des bus.

L'UAL et l'unité de contrôle forment le **processeur** ; on dit aussi **CPU** pour **C**entral **P**rocessing **U**nit.

2.2 L'unité arithmétique et logique

Elle est chargée d'effectuer le traitements des opérations arithmétiques ou booléennes :

— addition, multiplication, soustraction, division
— ET, OU et NON logiques ;
— décalages de bits dans des registres

L'UAL est entourée généralement de **registres de données** (*mémoires rapides*) et d'un **accumulateur** qui accueille les opérandes des opérations ou le résultat.

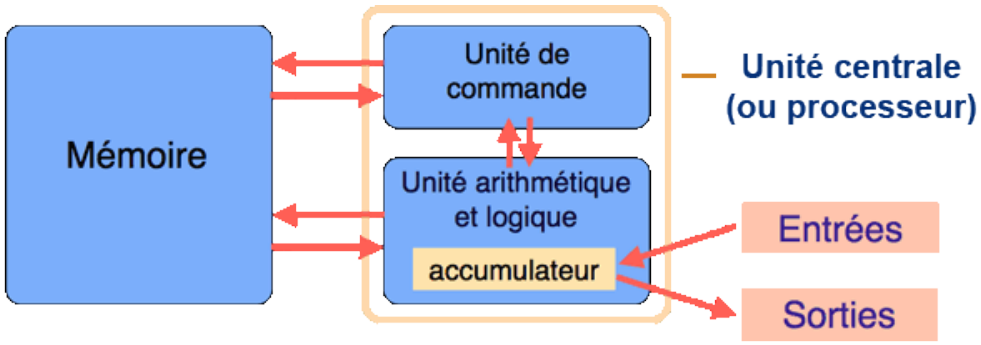

Fig. 2 – Modèle original

2.3 L'unité de contrôle

Elle est chargée de contrôler les échanges, gérer l'enchainement des instructions et les transferts entre les différents éléments.

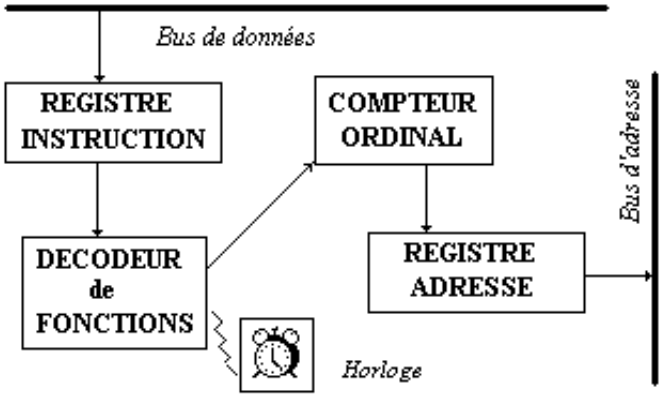

Fig. 3 – Unité de contrôle

On y trouve (*entre autres*) :

— un compteur ordinal ou « program counter PC »qui contient l'adresse de la prochaine instruction ;
— un registre d'instruction ou « instruction register IR ou current instruction register CIR »qui contient l'instruction lue ;
— un décodeur d'instruction ;
— un registre d'adresse ou « memory address register MAR »qui contient l'adresse de l'instruction à lire ;
— une horloge.

2.4 Les bus

Les différentes unités sont interconnectées par des systèmes de câblage appelé bus. Autour du processeur on trouve :

— le bus d'adresse unidirectionnel ;
— le bus de données bidirectionnel ;
— le bus de contrôle bidirectionnel.

Remarque : les ordinateurs récents possèdent d'autres types de bus.

2.5 Les instructions

Une instruction désigne un ordre donné au processeur. Il s'agit d'une chaine binaire de p bits composée de deux parties.

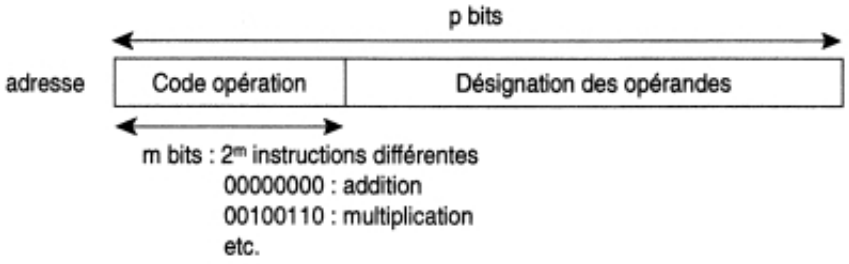

Fig. 4 – Format d'une instruction

Pour s'affranchir des codes binaires et des calculs d'adresse le programmeur utilise plutôt un **langage d'assemblage** où les instructions binaires sont remplacés par une chaine de caractères mnémoniques. Un programme appelé **assembleur** réalisera ensuite le passage vers le code binaire.

Les intructions du langage machine peuvent être rangées dans six catégories : calcul, transfert, entrées/sorties, saut, appel de sous programme, instructions particulières (arrêt par exemple). L'ensemble des codes opération reconnu par un processeur s'appelle son *jeu d'instructions*.

Il n'est pas question dans ce cours de détailler un quelconque jeu d'instructions, ni les subtilités de son utilisation. Il s'agit plutôt de présenter quelques séquences simples.

Dans les exemples qui suivront le jeu d'instructions minimaliste utilisé (*voisin de celui des processeurs ARM*) peut être trouvé à cette adresse. Les commentaires sont précédés du caractère ;.

2.5.1 Exemple 1

```
MOV R0,#30;chargement du registre R0 avec la valeur 30
MOV R1,#20
ADD R2,R1,R0;additionne les contenus de R0 et R1 et place le contenu dans le registre R2
STR R2,100;transfert le contenu de R2 à l'adresse 100
HALT;arret des traitements
```

2.5.2 Exemple 2

```
0       MOV R0,#30
1       MOV R1,#20
2       MOV R3,#0
3       ADD R2,R1,R0
```

```
4       CMP R2,#0
5 si:   BNE fin
6       MOV R2,R3
7 fin:  HALT
```

Dans cet exemple, on introduit la possibilité d'introduire des *étiquettes* utiles pour réaliser des sauts. Elles évitent d'avoir à calculer l'adresse de l'instruction en question (*laissant ce travail à l'assembleur*). La structure présentée ici est équivalente à un test sans alternatives.

2.5.3 E1C5 Que fait cette séquence d'instructions ?

1. Expliquer ce réalise les instructions ci-dessous :

```
MOV R0,#25
    STR R0,20
    MOV R1,#6
    STR R1,21
    ADD R0,R1,R0
    LSL R0,R0,#1
    STR R0,22
    HALT
```

2. Vérifier vos résultats avec le simulateur de Peter Higginson.

2.5.4 E2C5 Langage d'assemblage

Écrire en langage d'assemblage les instructions correspondant aux actions suivantes :

- ▷ comparer la valeur du registre R4 avec la valeur 18 (décimale)
- ▷ si celle-ci est plus grande alors sauter à l'étiquette 'etiqu1'
- ▷ charger la valeur 14 dans le registre R0 ;
- ▷ arrêter les traitements
- ▷ déclarer l'étiquette 'etiqu1'
- ▷ charger la valeur 18 dans le registre R0 ;
- ▷ arrêter les traitements

2.6 E3C5 Execution d'une séquence d'instructions

On utilise le simulateur de Peter Higginson.

1. Entrer les instructions suivantes (**sans les numéros de ligne !**) :

```
0       MOV R0,#30
1       MOV R1,#20
2       MOV R3,#0
3       ADD R2,R1,R0
4       CMP R2,#0
5 si:   BNE fin
6       MOV R2,R3
7 fin:  HALT
```

2. Executer cette séquence pas à pas (bouton **STEP**). Comment évolue le registre PC du processeur ?
3. Que se passe-t-il à la ligne 5 ? Expliquer l'évolution du registre PC.

2.7 Perspectives

Plus de 75 ans après sa présentation le modèle d'architecture de Von Neumann est toujours valable. Une différence peu significative par rapport au modèle original, est que les dispositifs d'entrées/sorties peuvent communiquer avec la mémoire par le biais de contrôleur dédié (DMA). Le schéma actuel serait plutôt le suivant :

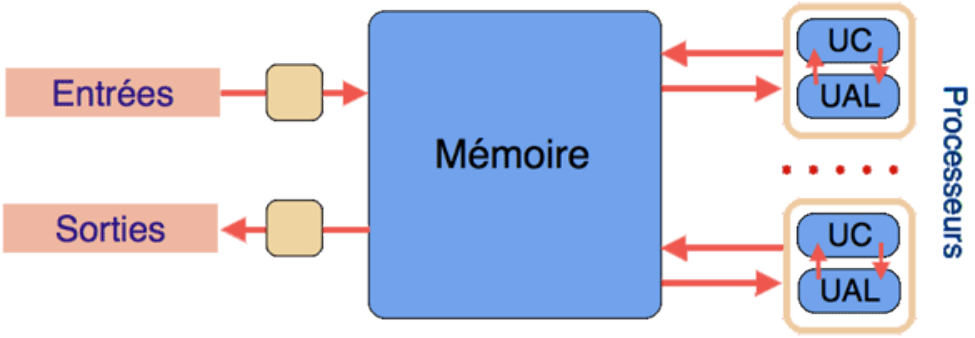

Fig. 5 – Modèle actuel

les ordinateurs actuels comportent plusieurs processeurs (on dit aussi plusieurs « cœurs ») intégrés sur une même puce. Cette tendance au « parallélisme »dans le traitement et la circulation des informations conduit à une augmentation de la puissance de calcul sans augmenter la fréquence des processeurs individuels.

3 Modèle d'architecture de Harvard

Dans le modèle d'architecture de Harvard les instructions et les données sont situées dans des mémoires différentes et sont véhiculés sur des bus indépendants.

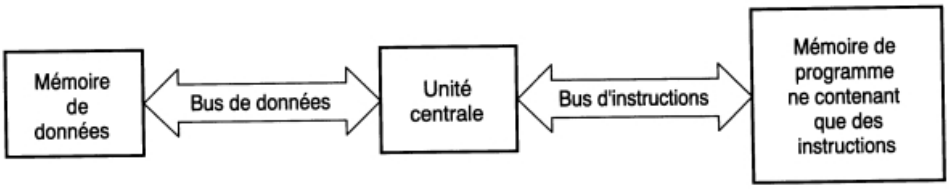

Fig. 6 – Modèle d'architecture de Harvard

La vitesse d'execution est de fait améliorée car en un seul cycle d'horloge on peut récupérer les données et le code instruction.
L'architecture de Harvard se retrouve beaucoup dans les systèmes embarqués.