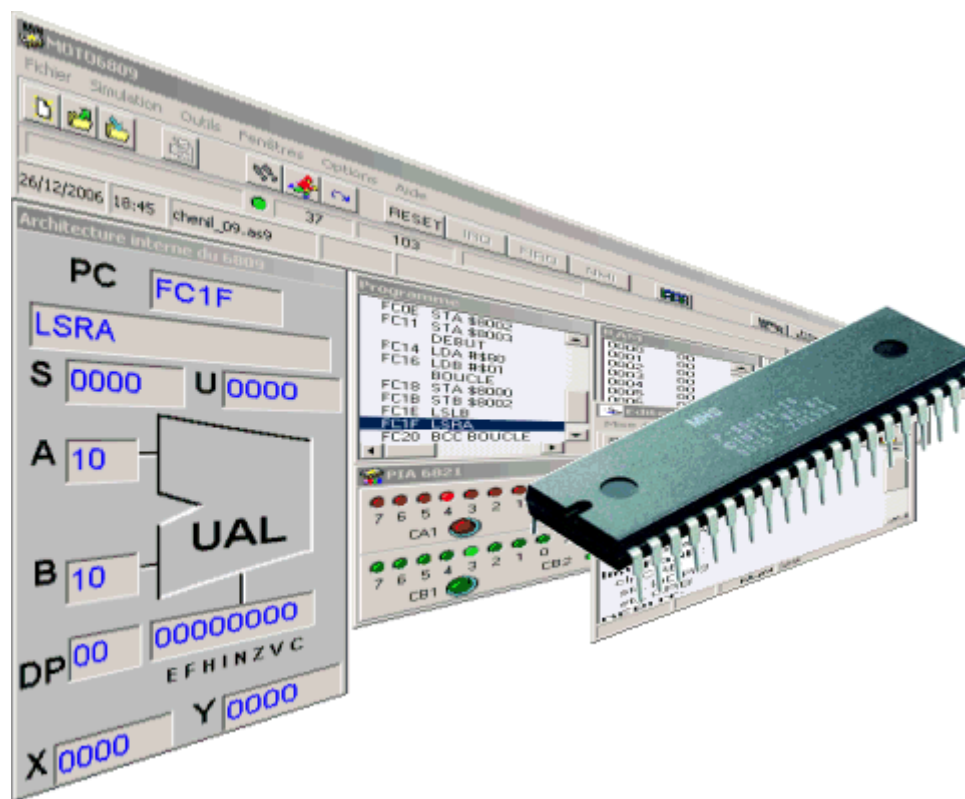


Documentation Technique

Simulateur de Microprocesseur Motorola

6809



Réalisé par: Youssef Bentigga-Abdellah Ouabi

Encadré par: Pr Hicham Benalla

Filière: Génie Informatique

Année universitaire: 2025–2026

I. Algorithme de compilation et Guide d'utilisation:

1.Introduction:

- Ce projet consiste à réaliser un simulateur du microprocesseur Motorola 6809. L'objectif est de permettre l'apprentissage, le test et le débogage des programmes.

2. Description du microprocesseur Motorola 6809:

- Le Motorola 6809 est un microprocesseur 8 bits introduit en 1978 ,il était le premier microprocesseur pour lequel il était possible d'écrire du code indépendant de sa position en mémoire et complètement ré-entrant d'une manière simple, sans utiliser d'artifice de programmation.

3. Algorithme de compilation:

- La compilation suit les étapes standards d'un projet Java suivantes:

1. Organisation du code source:

- cpu: émulation du cœur du 6809 (registres, instructions, flags)
- memory: gestion de la mémoire
- debugger: débogueur, points d'arrêt, pas à pas
- io: entrées/sorties et interruptions
- gui: interface graphique

2. Compilation:

- Utilisation du JDK (javac) sur Eclipse.

3. Création du fichier exécutable:

- Génération d'un fichier JAR

4. Exécution du fichier.

4. Guide d'utilisation:

- Pour utilisé le microprocesseur on suit les étapes suivantes:

1. Lancement du simulateur:

- On ouvre Eclipse, on clique sur fichier puis Double-clique sur le fichier

2. Interface utilisateur:

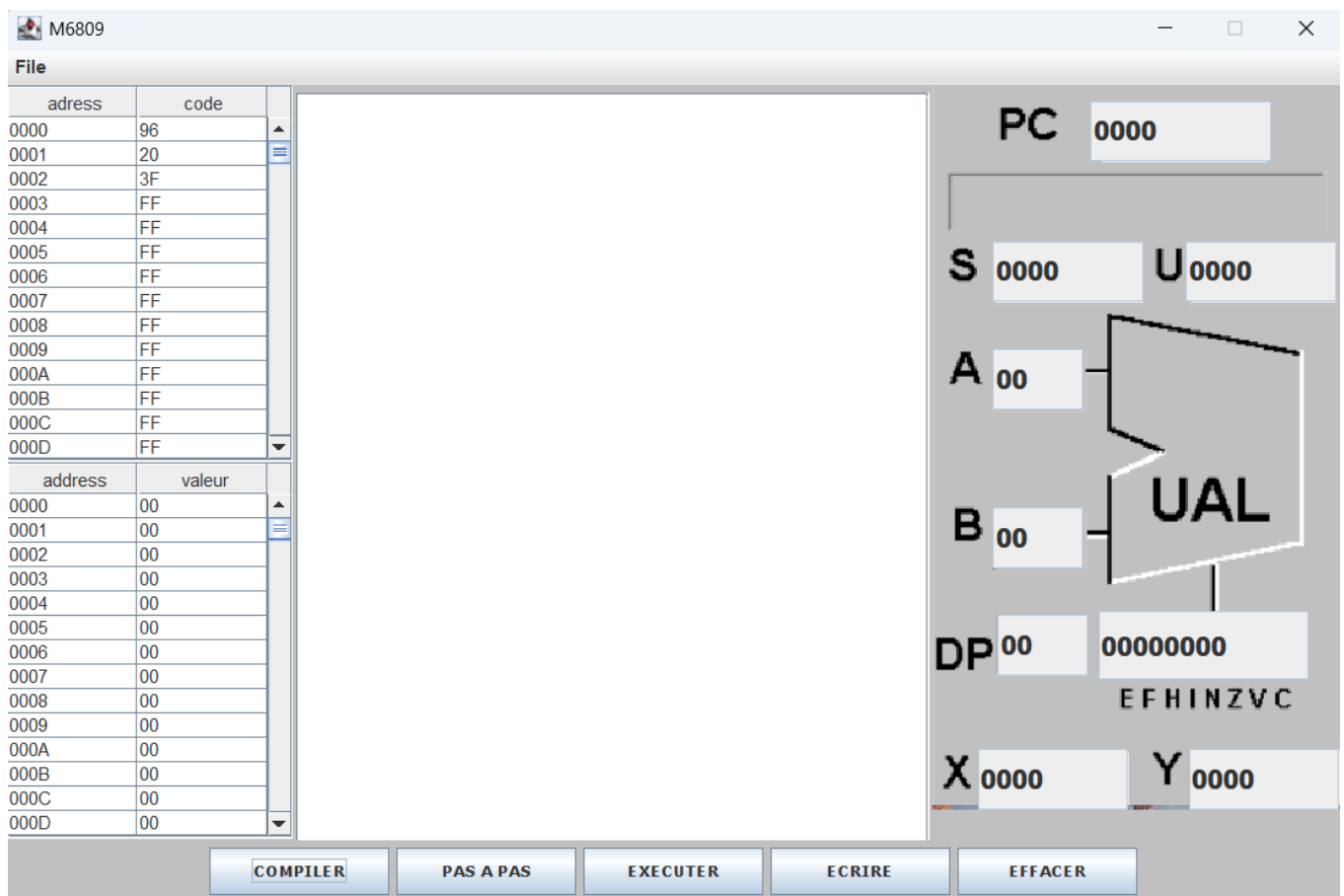
- Zone registres: affichage des registres A, B, DP, X, Y, U, S, PC, CCR
- Zone mémoire: visualisation hexadécimale
- Console: affichage des sorties

3. Débogage:

- Bouton Exécuter: lance le programme
- Bouton Pas à Pas: exécution instruction par instruction
- Bouton compiler: compilation de programme
- Bouton effacer: pour effacer le programme
- Bouton écrire: donner des valeurs à chaque adresse

4. Modification en temps réel:

- Modification manuelle des registres
- Modification du contenu mémoire

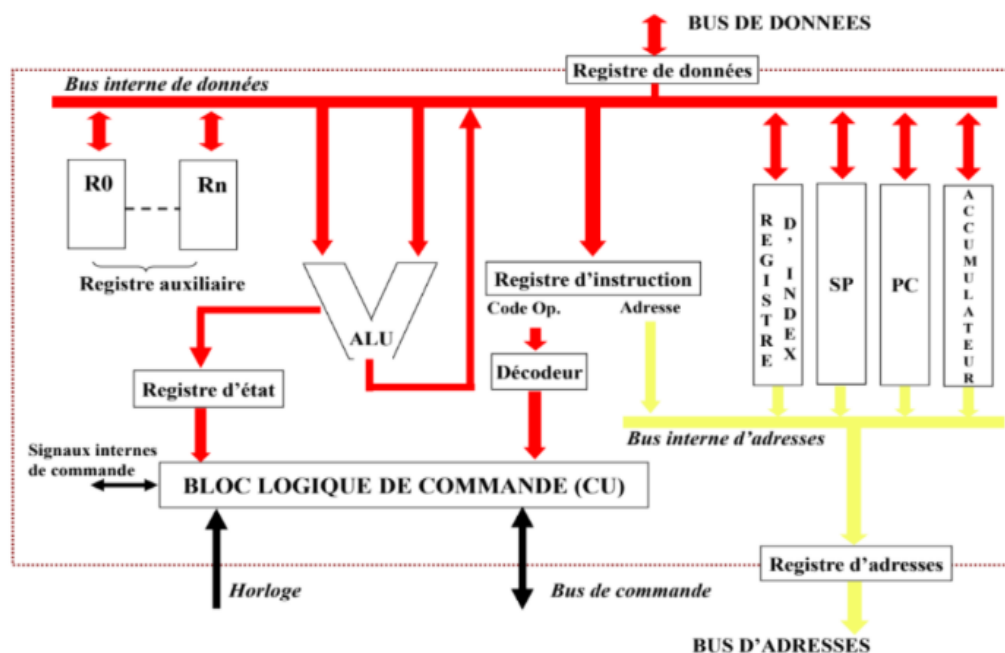


II. Documentation technique du fonctionnement du simulateur:

1. Architecture Interne du Simulateur:

- Le simulateur repose sur une architecture modulaire orientée objet. Chaque composant matériel du 6809 est représenté par une entité logicielle dédiée. Cette approche facilite la maintenance, l'évolution et le débogage.

➤ Architecture de base de microprocesseur:



- Bus interne de données: Transfère les données entre les différents blocs du processeur.
- Registres auxiliaires (R0 à Rn): sert à pour conserver des données temporairement.
- Registre d'état: Contient les informations sur l'état actuel des calculs.
- Unité arithmétique et logique(UAL): Réalise les calculs (addition, soustraction, etc.) et opérations logiques.
- Registre d'instruction: Stocke l'instruction en cours d'exécution.
- Décodeur d'instructions: Interprète l'instruction et détermine les actions à effectuer.
- Bloc logique de commande: Coordonne et contrôle les opérations du processeur.
- Bus interne d'adresses: Transfère les adresses entre les blocs.
- Registre d'adresses: Stocke l'adresse mémoire où le processeur doit lire ou écrire une donnée.

- Bus de données et bus d'adresses (extérieurs): Relient le processeur à la mémoire et aux périphériques externes.
- SP: Gère la pile (stack) utilisée pour le stockage temporaire.
- PC (compteur ordinal): Contient l'adresse de l'instruction suivante.
- Accumulateur: Sert pour les calculs et le stockage temporaire des résultats.

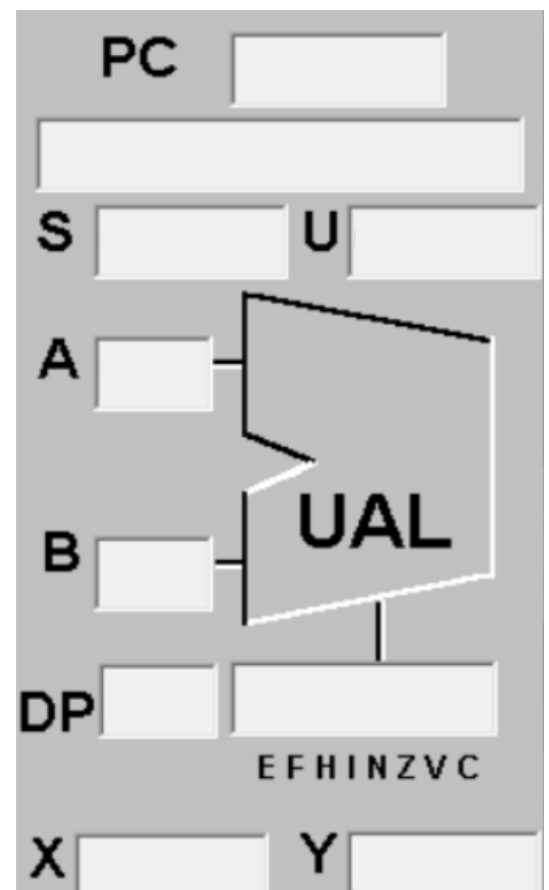
➤ Les principaux modules:

- CPUCore: cœur du processeur et cycle d'exécution
- InstructionDecoder: décodage des opcodes
- AddressingModeManager: gestion des modes d'adressage
- MemoryManager: mémoire RAM 64 Ko
- DebugEngine: moteur de débogage
- IOManager: entrées/sorties et interruptions
- GUIControlle: interface graphique

2. Émulation du CPU Motorola 6809:

➤ Les registres principaux du fonctionnement de notre microprocesseur:

- A, B: Accumulateurs 8 bits
- D: Registre 16 bits (A+B)
- X, Y: Registres index 16 bits
- U: Pointeur de pile utilisateur
- S: Pointeur de pile système
- PC: Compteur ordinal
- CCR: Registre d'état (flags N, Z, V, C, I, F, E)



➤ Etape d'exécution d'une instruction:

- Fetch: Lecture de l'opcode à l'adresse PC et Incrémentation du PC.
- Decodage: Identification de l'instruction, détermination du mode d'adressage et calcul des operands.
- Execute: Exécution de l'opération, mise à jour des registres et des flags CC.

3. Jeu d'instructions:

- Notre microprocesseur inclus des opérations arithmétiques, logiques et de manipulation de bits paramétrés par des modes d'adressage gérés par des méthodes permettant le calcul précis de l'adresse effective.

Les modes d'adressage pris en charge:

- Inhérent
- Immédiat
- Direct
- Étendu direct/indirect
- Indexé direct/indirect

- Après chaque instruction, CCR est mise à jour.

4. Gestion de la mémoire:

- La mémoire du 6809 est simulée par un tableau de 65536 octets. Chaque accès mémoire est contrôlé afin de permettre la surveillance en débogage.
- Les méthodes read(address) et write (address, value) assurent la validation des adresses et la détection des erreurs.

5. Moteur de débogage:

- Le moteur de débogage est intégré directement au cycle d'exécution du CPU, il permet un contrôle total de l'exécution.
- Points d'Arrêt: Les utilisateurs peuvent arrêter l'exécution à des endroits précis pour examiner le programme.
 - Pas à pas: Exécution des instructions une par une pour mieux comprendre le programme.

6. Gestion des entrées et sorties:

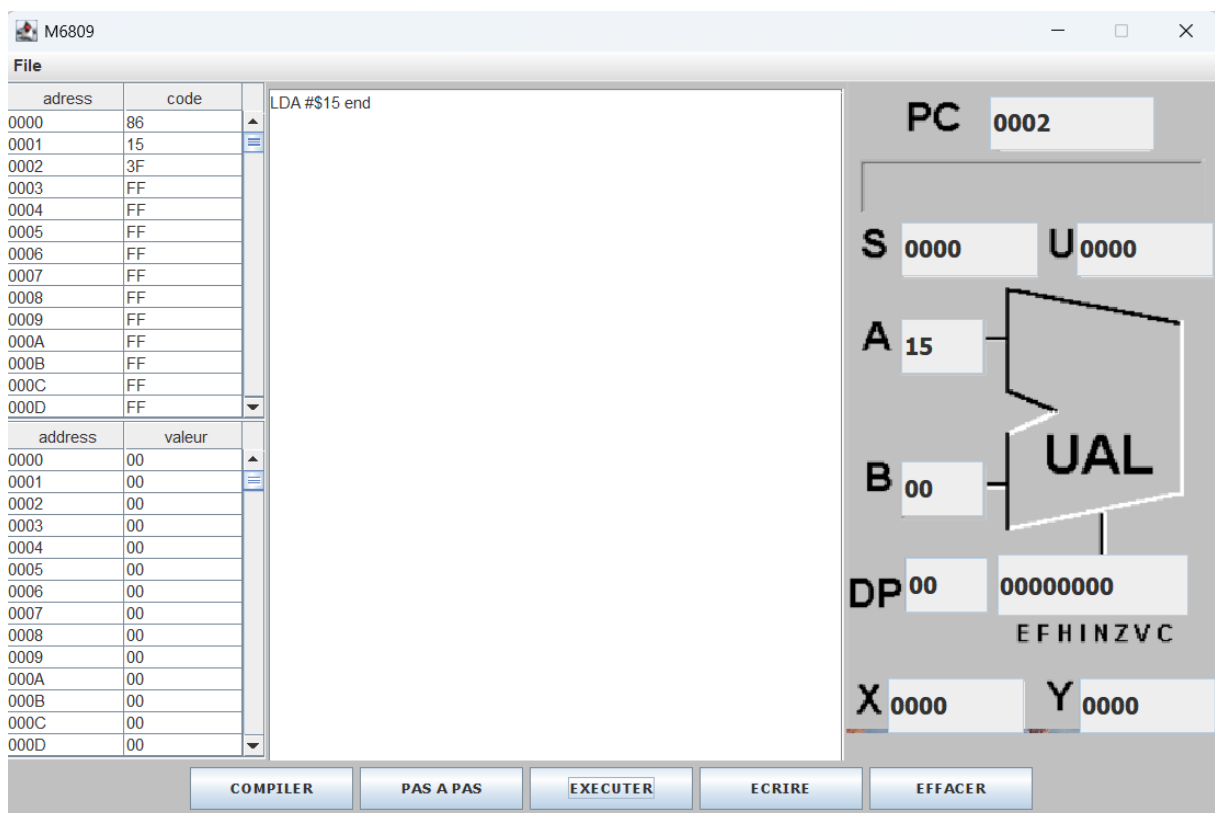
- Les entrées sont simulées par des périphériques comme le clavier et la souris.
- Les sorties: Affichage des résultats du programme sur une console ou une interface graphique.

7. Interface Graphique:

- L'interface graphique est développée avec Swing sur Eclipse. Elle communique avec le cœur du simulateur via des observateurs.

8. Test du fonctionnement de notre microprocesseur:

- Pour ce la on va écrire notre instruction avec un mode d'adressage de notre choix pour vérifier le fonctionnement M6809 comme le montre la capture ci-dessous:



- On remarque que notre microprocesseur marche très bien ,il a chargé la valeur 15 dans l'accumulateur A.

III.Conclusion:

Ce projet avait pour objectif principal la conception et la réalisation d'un simulateur du microprocesseur Motorola 6809. À travers ce travail, on a créé les mécanismes internes d'un microprocesseur avec précision, notamment la gestion des registres, le cycle d'exécution Fetch–Decode–Execute, le jeu d'instructions ainsi que les différents modes d'adressage spécifiques pour le fonctionnement de notre microprocesseur 6809.