

R204 – Projet Communication Bas Niveau

PAYOL Albert
BEN OUIRANE TAYCIR



Table des matières

1.	Introduction	3
2.	Création des Champs	3
3.	Conception des Instructions	4
4.	Implémentation des Programmes	5
F	Programme 1 : Vérification du Palindrome	5
F	Programme 2 : Conversion en Majuscules	6
5.	Conclusion	6



1. Introduction

Ce rapport présente le travail réalisé dans le cadre d'un projet en binôme visant à développer un algorithme en utilisant un ensemble d'instructions spécifiques conçues à l'aide du logiciel MAG. Notre objectif principal était de concevoir un ensemble d'instructions et de champs permettant d'effectuer des opérations bas niveau sur une architecture de processeur hypothétique.

Le domaine du bas niveau en informatique est crucial pour la compréhension et la manipulation directe du matériel informatique. Il implique des opérations proches du langage machine, offrant un contrôle précis sur les composants matériels tels que les registres, la mémoire et les opérations d'entrée/sortie.

Le logiciel MAG (Microprogrammable Architecture Generator) est un outil puissant utilisé par les concepteurs de processeurs pour simuler, tester et développer des architectures de processeurs personnalisées. Il permet aux ingénieurs de définir des jeux d'instructions sur mesure, des champs de contrôle et des opérations spécifiques pour créer des processeurs adaptés à des applications spécifiques.

2. Création des Champs

Nous avons initié le projet en concevant plusieurs champs dans le logiciel MAG. Voici une liste des champs créés avec leurs attributs :

- JMP
- MUX1
- MUX0
- MUX2
- DEC
- ALU
- DATA
- JMPZ
- JMPNZ JMPN
- JMPPZ

Ces champs sont organisés sur 32 bits, où les 16 premiers bits représentent les données, et les 16 bits suivants sont réservés pour les différents champs.



3. Conception des Instructions



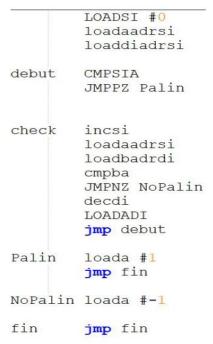
Nous avons élaboré les instructions en utilisant un tableau Excel pour spécifier les différents champs et leurs valeurs associées. Voici un résumé des principales instructions et leurs fonctionnalités :

- LOAD_A #VAL: Charge une valeur immédiate dans le registre A en utilisant le décodeur et l'ALU.
- LOAD_SI #VAL: Charge une valeur immédiate dans le registre SI.
- LOAD_A_DI: Stocke l'adresse de DI dans le registre A.
- **INC** et **DI**: Permettent de récupérer la valeur d'une variable/pointeur (SI/DI).
- **CMP**_#_#: Instructions de comparaison utilisant l'ALU pour comparer deux valeurs.
- **JMP...**: Instructions de saut conditionnel utilisant les champs JMP correspondants et l'opérande ADRESSE DE SAUT.
- LOAD_A_SI: Récupère la valeur de SI et la stocke dans le registre A.
- LOAD_B_ADR_SI: Charge la valeur pointée par SI dans le registre B.
- LOAD_ADR_SI_B: Copie le registre B dans la case pointée par SI.



4. Implémentation des Programmes

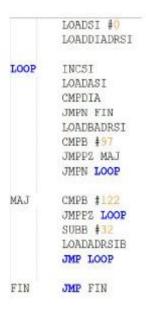
Programme 1 : Vérification du Palindrome



Ce programme commence par initialiser le pointeur SI à 0, puis utilise une boucle pour comparer les caractères aux extrémités de la chaîne. Si les caractères ne correspondent pas, le mot n'est pas un palindrome. Sinon, le programme continue jusqu'à ce que les pointeurs SI et DI se croisent.



Programme 2 : Conversion en Majuscules



Dans ce programme, chaque caractère de la chaîne est converti en majuscule si nécessaire. Le programme parcourt la chaîne caractère par caractère, vérifie si le caractère est en minuscule, et le convertit en majuscule si nécessaire en ajustant son code AS-CII.

5. Conclusion

Ce projet a été une expérience précieuse pour comprendre le fonctionnement interne d'un processeur et la logique sous-jacente à la programmation bas niveau. En concevant des champs, des instructions et des programmes, nous avons acquis une compréhension approfondie des concepts fondamentaux de l'architecture informatique.