Mini projet : Calculatrice basique

Architecture - TP

I. Introduction :

Le but de ce mini projet était de réaliser une calculatrice basique en assembleur MIPS. Le système va orienter le programme en fonction du mode demandé (integer par défaut). Le système demande ensuite une opérande et un opérateur à l'utilisateur. Si l'utilisateur entre un opérateur qui n'a besoin que d'une seule opérande, alors il affiche le résultat. Sinon il demande une deuxième opérande à l'utilisateur et affiche le résultat. Dans les deux cas, le calcul est immédiatement effectué. Le système remonte, affiche le résultat sur une nouvelle ligne et considère alors le résultat comme étant la première opérande et demande un nouvel opérateur.

II. Les fonctionnalités de base:

La calculatrice peut effectuer des calculs dans trois modes (integer, float et double) qui ne possèdent pas forcément les mêmes fonctionnalités (voir chapitre suivant). Dans n'importe quel mode disponible, à partir de deux opérateurs, la calculatrice peut effectuer des additions « **+** », des soustractions « **-** », des multiplications « ***** » et des divisions « **/** ». Elle peut calculer la puissance « **pow** » (x^a avec $x \in \mathbb{R}$, $a \in \mathbb{Z}$) et comparer deux nombres avec « **min** » et « **max** ». La calculatrice ne nécessite qu'une seule opérande pour calculer la valeur absolue « **abs** ».

III. Les fonctionnalités supplémentaires :

- Mode double : A l'identique du mode float, mais avec une meilleure précision (double).
- Changement de mode : Si l'utilisateur entre le nom d'un mode en opérateur (integer - float - double), alors on saute dans ce mode et on garde la première opérande.
- Fin : Si l'utilisateur entre une ligne vide en opérateur, alors on ferme le programme.
- Reset du résultat : Si l'utilisateur entre « reset » en opérateur, alors on réinitialise le résultat à zéro et on boucle pour demander un nouvel opérateur.
- Calcul de l'opposé : on retourne la négation de la première opérande.
- Puissance négative : on calcule la valeur absolue de la deuxième opérande et on retourne la division de 1 par le calcul d'une puissance classique.

Fonctions spéciales pour integer :

- Binaire : Affichage de 32 bits, un à un, grâce aux instructions « sll » et « srl ».
- Hexadécimal : On convertit 4 bits par 4 bits (via la table ASCII) qu'on stocke dans un string, on l'affiche et on quitte le programme.
- PGCD : On identifie le maximum et le minimum, puis on utilise les quotients et restes.
- Premier : On retourne le nombre si il est premier, sinon on retourne zéro.
- Factoriel : Calcul de la suite factorielle au rang de la première opérande.
- Fibonacci : Calcul de la suite de Fibonacci au rang de la première opérande.
- Modulo : Reste de la division des deux opérateurs.
- Multiplication : J'ai ajouté la multiplication via le binaire avec « sll » et « srl ».

Fonctions spéciales pour float / double:

- Calcul de l'inverse : si l'opérande est nulle alors on initialise le résultat à zéro. Sinon on divise un par l'opérande et on retourne le résultat.
- Exponentielle : Calcul de e^x avec x la première opérande.
- Logarithme népérien : Calcul de ln(x) avec x la première opérande.
- Racine carrée : Calcul de la racine carrée via « sqrt.s » ou « sqrt.d »
- Arrondi : Le système convertit la première opérande en entier avant de la reconvertir dans le mode de base.

IV. Elaboration du projet :

Mes premières versions n'utilisaient pas le code proposé sur Moodle et ne fonctionnaient donc pas avec des paramètres. J'utilise maintenant le code suggéré et il est à présent possible de lancer la calculatrice avec des paramètres depuis le terminal.

J'ai commencé par développer le mode integer. C'est dans cette partie que j'ai eu le plus de difficultés, notamment au niveau de la structuration du code mais aussi à trouver une méthode pour comparer des opérandes pré-enregistrées avec l'opérande entrée par l'utilisateur. Une fois terminé, j'ai codé les fonctions permettant d'exécuter les opérations de base et j'ai ajouté des sauts pour permettre au système de pouvoir boucler.

Une fois le mode integer achevé, le schéma était globalement le même pour les autres modes (float et double) : je l'ai répété, puis adapté au mode concerné et j'y ai ajouté des fonctions qui permettent de boucler avec le mode demandé. J'ai surtout rencontré des difficultés au niveau de l'adaptation du code au mode demandé, à cause de la gestion des registres et du Coprocesseur 1.

Une fois que les opérations basiques fonctionnaient, j'ai commencé à développer les fonctionnalités supplémentaires suggérées dans le sujet. J'ai commencé par le changement de mode, puis les calculs d'inverse et d'opposé. Ensuite, j'ai débuté le binaire et l'hexadécimal. J'ai eu beaucoup de difficultés lors du codage de ces fonctions de conversions et c'est à partir d'ici que j'ai continué avec le code de Moodle.

Puis j'ai eu l'idée d'ajouter d'autres fonctionnalités au mode integer : le pgcd, la suite factorielle, la suite de Fibonacci, le modulo, la multiplication via le binaire et dire si un nombre est premier ou non. J'ai fini en ajoutant d'autres fonctionnalités, cette fois au mode float et double : l'exponentielle, le logarithme népérien, la racine carrée et l'arrondi.

J'ai décidé de ne pas ajouter les fonctionnalisés du mode integer (binaire, hexadécimal, pgcd, premier, factoriel, Fibonacci, modulo) au mode float ou double car les résultats n'ont pas besoin d'être des flottants. De même pour les fonctionnalités du mode float ou double, je ne les ai pas ajouté au mode integer car le résultat est arrondi et est donc trop imprécis.

J'ai également tenté de coder les puissances flottantes, mais je n'ai pas compris comment le faire d'un point de vue mathématiques, ce qui est dommage car j'aurai ainsi pu coder la racine carrée d'une autre manière. Je n'ai pas non plus pu ajouter la multiplication binaire aux autres fonctionnalités ou modes. Il y a aussi un soucis après l'affichage du string après la conversion en hexadécimal que je n'ai pas réussi à résoudre.

V. Remerciements :

Afin de réaliser ce projet, j'ai utilisé le simulateur MARS. Pour le codage, j'ai regardé la série de 38 vidéos de « Amell Peralta » sur YouTube qui portait sur l'assembleur MIPS. Je me suis également renseigné sur le site stackoverflow. (Voir liens dans la partie **Annexes**).

VI. Conclusion :

J'ai apprécié de développer ce mini-projet de calculatrice basique. En effet, j'ai pu mieux comprendre le fonctionnement de l'assembleur et la gestion des adresses. Ce projet m'a également permis d'avoir un aperçu du monde professionnel étant donné qu'il nous était donné un objectif à atteindre dans un temps imparti. J'estime avoir correctement répondu au sujet et je suis fier de ce que j'ai fait. Je suis tout même déçu de ne pas avoir réussi à coder les dernières fonctionnalités indiquées à la fin du grand 4. J'espère pouvoir y remédier par la suite. L'envisage aussi d'ajouter d'autres fonctionnalités comme pouvoir afficher la mantisse d'un nombre flottant ou afficher l'exposant d'un nombre flottant (fonctionnalités suggérées dans le sujet mais que je n'ai pas pu mettre en oeuvre dans ce projet).

VII. Annexes :

- Amell Peralta : https://www.youtube.com/channel/UCPZ473Q4kbG98JmL71PgXTA
- Stackoverflow : https://stackoverflow.com