Alicia CALMET Année 2020-2021  
Loïc ROBERT 4 IR A2

COMPTE-RENDU  
PROJET SYSTEME INFORMATIQUE

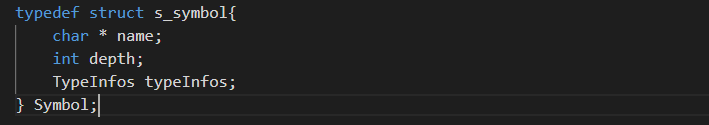
Lien github partie compilateur :  
Lien github partie microprocesseur :

**Démarche de conception :**

Tout au long de projet, nous avons suivi le schéma organisationnel proposé sur la page Moodle. De plus, afin de ne pas être en retard sur le planning des séances, nous avons travaillé en dehors des heures de cours. Ainsi, notre projet a abouti et nous avons pu implémenter de nouvelles fonctionnalités qui n’étaient pas attendu à notre compilateur.

**Partie Compilateur en utilisant LEX et YACC:**

***Analyse lexicale :*** l’analyseur lexical détecte les mots typés (tokens) spécifiques à notre langage. Ce fichier (compiler.l) est séparé en trois parties : une partie est consacrée aux définitions, une partie est consacrée aux règles et une dernière partie est dédiée au code. La partie LEX lit donc le fichier, reconnaît et décode les tokens définis et les renvoie à la partie YACC que nous allons aborder maintenant.

***Analyse syntaxique***: l’analyseur syntaxique analyse notre langage de type C et produit comme sortie le code assembleur correspondant. Le fichier (compiler.y) se structure de la même façon que la fichier compiler.l. Le seul changement est que dans YACC, la partie règles définit l'ensemble des règles qui définissent notre grammaire au lieu des expressions régulières dans LEX.  
Lorsque nous codons, nous utilisons des variables. Ainsi, nous avons créé une table des symboles et des fonctions déclarer dans symbol.h qui permettent de les gérer. Un symbole a un nom, une profondeur (les variables globales sont à une profondeur de 0, celles déclarés dans la main à une profondeur 1, etc), et un type ().  
Nous avons aussi implémenté une table des symboles temporaires qui fonctionne comme une pile pour les calculs.

Toutefois, au-delà de l’implémentation des expressions conditionnelles *if* et *while* demandée, nous avons implémenté de nouvelles fonctionnalités qui n’étaient pas attendu à notre compilateur.

En effet, nous avons implémenté :  
- la possibilités d’écrire des commentaires (ils sont gérés de la même manière que les espaces)  
- l’analyse d’une condition dans laquelle il y a une expression à droite et/ou à gauche

Une image contenant texte, horloge

Description générée automatiquement  
- le test que l’on n’affecte pas une valeur nouvelle a une constante  
- la possibilité de déclarer des variables sur une même ligne en les séparant par une virgule/ la possibilité d’affecter des valeurs à des variables sur une même ligne en les séparant par une virgule

  
- le parsage du type d'une variable : nous avons réalisé du typage fort  
- la gestion des opérateurs logiques ≥,<= et != en plus des opérateurs <,> et ==

- l’ébauches de pointeurs  
- l’affichage d’un message d’erreurs (warning) si un pointeur est au-dessus de la tête.  
-fprintf dans stderr pour warnings et erreurs

**Une image contenant texte

Description générée automatiquement**  
- la traduction avec instruction en code instruction

**Partie Microprocesseur de type RISC:**