POILLY Kilyan

000460014

B1-INFO

Projet annuel de Fonctionnement des ordinateurs :

- Présentation du projet :

  Le but de ce projet était de simuler le fonctionnement d'un processeur possédant 4 registres de travail (R0, R1, R2 et R3), et 3 registres de contrôle (IR, PC et ST), ici, j'ai décidé de les regrouper dans un dictionnaire nommé Registers, cela me paraissait être la solution la plus simple et la plus pratique lorsqu'il s'agissait de changer la valeur de contenue dans ceux-ci.

  Ce processeur « virtuel » est associé à une mémoire primaire contenant des instructions et des valeurs, que j'ai également créé sous forme d'un dictionnaire nommé Memory, cela me permettait de créer, par exemple, une case mémoire « 42 » en ayant une mémoire plus courte, par exemple, si je l'avais fait sous forme de liste, si la liste avait 5 composants mais que je voulais stocker une valeur à l'adresse 42, j'aurai très probablement obtenu une erreur « index out of range ».

  Chaque valeur ou instruction utilisée dans par ce processeur virtuel sont stockées en complément à 2 afin de pouvoir gérer les valeurs négatives.

  Voici les différentes fonctions que j'ai mis en place afin d'arriver au résultat demandé :

- Les fonctions :

  InitMemory :
  Cette fonction permet d'initialiser la mémoire, elle va donc chercher des informations dans 2 fichiers distincts et les stocks dans la mémoire, le premier fichier contient les instructions, le second contient des valeurs à stocker dans les cases mémoire associées.

  Fetch :
  Cette fonction simule l'étape « Fetch » d'un processeur, elle stock l'instruction correspondant au nombre PC dans le registre IR, et incrémente PC.

  Decode :
  Cette fonction simule l'étape « Decode » d'un processeur, elle détermine l'OPCode et les paramètres donnés par l'instruction stockée précédemment dans IR.

  Execute :
  Cette fonction simule l'étape « Execute » d'un processeur, elle exécute l'instruction correspondant à l'OPCode sur les registres ou valeurs données en paramètre.

  RegistertoRegister :
  Cette fonction est utilisée dans le cas où l'instruction en cours est effective d'un registre à un autre, elle return alors 2 Registres données par les paramètres.

RegistertoMemAd :
Cette fonction est utilisée dans le cas où l'instruction en cours utilise une adresse mémoire et un registre, elle return alors l'adresse mémoire et le registre données en paramètres.

Valtodest :
Cette fonction est utilisée dans le cas où l'instruction en cours utilise le paramètre en tant que valeur à stocker dans un registre donné par l'instruction, elle return alors la valeur, la registre de destination est donné par l'instruction.

Binarytodecim :
Cette fonction permet de convertir un nombre donné en complément à 2, et le return en décimal.

ChangeST1stbit :
Cette fonction change le premier bit du registre ST en a ('1' ou '0') selon le résultat de l'opération effectuée en ADD

WRL :
Correspond à l'instruction du même nom.
« Écrit la valeur donnée par les 4 bits de paramètre dans les bits de poids faible du registre R0 .»

WRH :
Correspond à l'instruction du même nom.
« Écrit la valeur donnée par les 4 bits de paramètre dans les bits de poids fort du registre R0. »

MOV :
Correspond à l'instruction du même nom.
« Copie une valeur d'un registre source à un registre destination. »

STR :
Correspond à l'instruction du même nom.
« copie le contenu d'un registre numéro à l'adresse mémoire donnée par un registre destination. »

RD :
Correspond à l'instruction du même nom.
« Cette instruction est symétrique de STR. L'instruction copie, dans un registre destination, le contenu de la cellule mémoire dont l'adresse est donnée dans le registre source. »

ADD :
Correspond à l'instruction du même nom.
« . Cette instruction stocke dans le registre destination la somme des valeurs contenues dans les registres destination et source. Cette instruction met également le bit 0 du registre ST à 1 si et seulement si le résultat de l'opération est 0.»

INV :
Correspond à l'instruction du même nom.
« Remplace le contenu du registre donné par son opposé. »

JP :
Correspond à l'instruction du même nom.

« Instruction de saut. Cette instruction ajoute au registre PC la valeur donnée par les 4 bits de paramètre. »

JZ :
Correspond à l'instruction du même nom.
« Instruction de saut conditionnel. Cette instruction ajoute au registre PC la valeur donnée par les 4 bits de paramètre si et seulement si le bit 0 du registre ST contient 1. »

JNZ :
Correspond à l'instruction du même nom.
« Cette instruction ajoute au registre PC la valeur donnée par les 4 bits de paramètre si et seulement si le bit 0 du registre ST contient 0. »

END :
Correspond à l'instruction du même nom.
« Arrête l'exécution. »

Print :
Fonction qui affiche à l'utilisateur l'avancée du programme, en affichant l'état (Fetch, Decode ou Execute), le cycle et les valeurs contenues dans les différents registres.

UI :
Fonction qui gère l'affichage de l'interface utilisateur, les changement d'état / de cycles ainsi que les différents choix qui s'offrent à l'utilisateur.