Travaux Pratiques d'Assembleur

Les TP d'Assembleur vous permettront de programmer le simulateur du microprocesseur présenté en cours et en TD. Vous devez faire valider l'avancée de votre travail par l'encadrant au moins une fois par séance et systématiquement à la fin de chaque exercice.

Tous les codes devront être indentés et suffisamment commentés pour être compréhensibles.

Initiation à la programmation du simulateur

Exercice 1: Fiche de cours à rendre en rentrant dans la salle

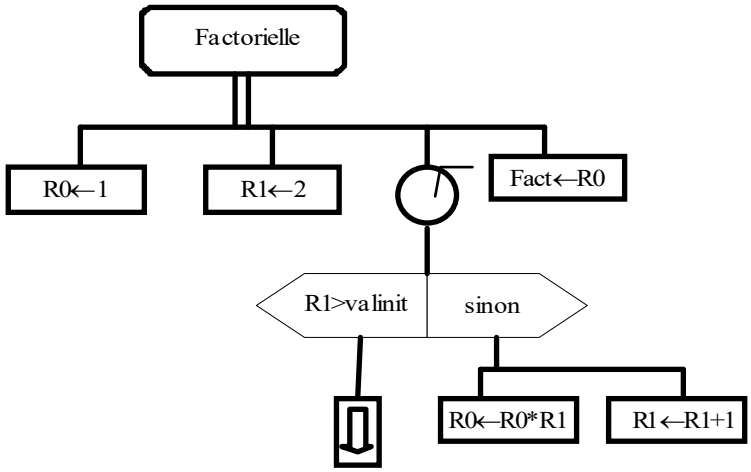
1. Parcourez le guide de programmation du simulateur de processeur en particulier le dernier paragraphe intitulé « Utiliser le simulateur ».
2. A partir des fichiers du cours et du guide de programmation, concevez un document texte d'une page résumant les informations essentielles nécessaires à vos débuts dans la programmation avec le simulateur.

Exercice 2: Tests des fichiers de démonstrations

1. A partir du cours d'elearn, copiez sur votre espace de travail le cours, le guide de programmation, ce sujet de TP, le fichier de l'exécutable du simulateur de microcontroleur et le répertoire des fichiers de démonstration. Lancez le simulateur en double cliquant sur l'exécutable ou à l'aide de la fenêtre de commande DOS.
2. Chargez le programme balle.mpc puis exécutez-le.
3. Rechargez le programme. A partir de l'affichage de la zone du microprocesseur dans le simulateur, identifiez à quelle adresse mémoire se trouve la première instruction du code.
4. Visualisez cette première instruction dans la mémoire. Quel code a écrit le programmeur pour obtenir cette instruction en mémoire ? Déterminez l'adresse de la pile et visualisez-la.
5. Lancez le programme en mode pas à pas avec une vitesse d'exécution faible et suivez l'évolution des registres et de la mémoire en fonction des instructions en cours de réalisation
6. Que contient SP à la fin de la première instruction ? Pourquoi ?
7. Vérifiez le fonctionnement des 4 premières instructions du programme en les exécutant pas à pas à vitesse faible puis exécutez normalement le programme.
8. Expliquez pourquoi Mem(101)=14 à la fin de ces 4 instructions. (voir diapositive 98 du cours pour le comprendre le fonctionnement du PUSH)
9. Testez les deux autres programmes de démonstration en les exécutant en mode pas à pas sur quelques instructions et en suivant l'évolution des registres et de la mémoire en fonction des instructions en cours de réalisation. Testez-les en vitesse normale.

Exercice 3: Factoriel

L'objectif de cet exercice est d'écrire un programme en assembleur qui calcule la factorielle de 6. Il respecte l'algorithme suivant :



Ce programme déclare une variable 'valinit' initialisée à 6 et une variable 'Fact' non initialisée.

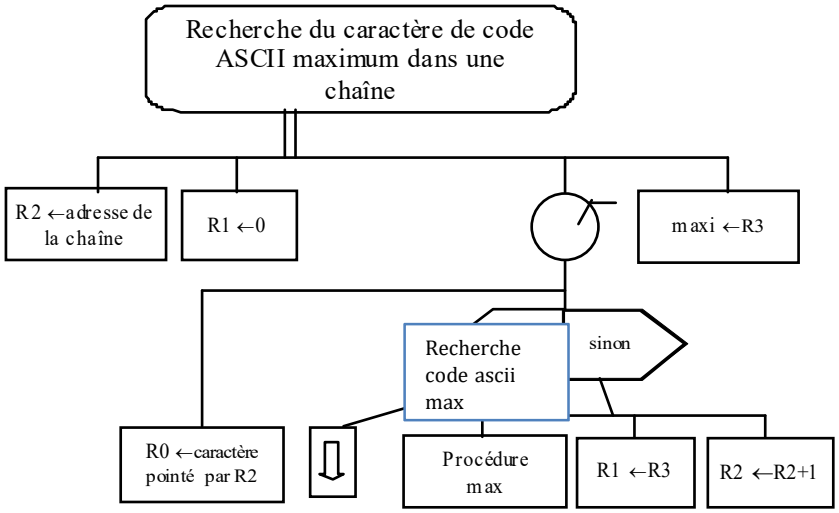
Vous pourrez utiliser un traitement de texte quelconque ou un EDI (wordpad, bloc note, Crimson Editor…).

1. Identifiez le test qui vous permettra de valider le programme réalisant cet algorithme.
2. Ecrivez le programme en assembleur. Si vous utilisez le Bloc Note veillez à ce que l'extension soit correcte si besoin en décochant « masquer l'extension des fichiers connus » dans Options des dossiers de l'explorateur.
3. Lancez le simulateur et compiler ce programme. Lorsque la compilation est réussie, charger le programme en mémoire. Ne pas oublier de choisir pour la variable 'Fact' un affichage en entier naturel (en cliquant sur sa valeur en mémoire pour accéder au choix du mode d'affichage), puis exécuter le programme en pas à pas de façon à voir comment évolue le contenu de la variable 'Fact'.
4. Copiez le programme dans un nouveau fichier que vous renommerez puis modifiez ce nouveau programme pour que la variable 'valinit' soit initialisée à 13. Exécutez le programme. Le résultat obtenu est-il correct ? Pourquoi ? Identifiez à quel moment et de quelle manière le simulateur vous indique ces informations.

Exercice 4: Chaîne de caractères

1. Ecrire un programme qui:
   - déclare une chaîne de caractères avec un texte de votre choix
   - recherche dans cette chaîne le caractère ayant le code ASCII le plus grand
   - place celui-ci dans une variable appelée 'maxi'.

La chaîne de caractères sera terminée par 0 (pas le caractère '0' mais le chiffre 0). Le programme devra suivre l'algorithme ci-dessous :



2. Identifiez le test qui vous permettra de valider le programme réalisant cet algorithme.

3. Chargez ce programme en mémoire et choisissez pour la variable 'maxi' un mode d'affichage en caractère. Exécutez le programme et vérifiez que, lorsqu'il se termine, on a bien dans 'maxi' le caractère de la chaîne choisie le plus loin dans l'alphabet : pour cela, choisissez un affichage sous forme de nombre pour les caractères de la chaine et le résultat en cliquant sur Mode d'affichage dans la zone de visualisation de la Mémoire.