

Travaux Pratiques d'assembleur

Les TP d'Assembleur vous permettront de programmer le simulateur du microprocesseur étudié en cours et en TD.

Le présent document vous présente :

- l'ensemble des exercices à réaliser lors des 5 séances de TP
- un calendrier de réalisation de ces exercices : à la fin de chaque séance de TP, vous devrez avoir terminé les exercices demandés. Si ce n'est pas le cas, il faudra les avoir terminés avant le début de la séance de TP suivante.

Vous devez faire valider l'avancée de votre travail par l'encadrant au moins une fois par séance et lui faire évaluer les exercices au fur et à mesure que vous les terminez.

Pour être validé et évalué un exercice doit remplir les conditions suivantes :

- **Tous les exercices précédents doivent avoir été validés et notés.** Les exercices sont donc à réaliser dans l'ordre.
- Vous êtes capable de répondre aux questions de l'exercice sans regarder vos notes (excepté pour les résultats de calcul).
- Vous êtes capables de justifier vos réponses.
- Vos codes sont indentés.
- Vos codes sont suffisamment commentés pour être compréhensibles par une tierce personne sans votre aide.
- Vous êtes capables de justifier chaque ligne de code.

Si toutes ces conditions ne sont pas remplies, l'exercice ne sera ni validé ni évalué même si votre code fonctionne. Si vous proposez un code que vous ne savez pas justifier, la note sera de 0 à l'item.

La validation et l'évaluation seront individuelles. Le barème prévisionnel et le calendrier de réalisation sont les suivants :

Partie	Exercice			N° de la séance de TP
	N°	Intitulé	Barème	
Initiation à la programmation du simulateur	1	Fiche de cours	0,25	1
	2	Test des fichiers de démonstration	1,75	
	3	Factoriel	2,50	
	4	Recherche de caractères	1,75	2
Affichage de texte	5	Affichage de texte	2,50	
Utilisation des touches et de la souris	6	Dessin conditionné par l'utilisateur	1,50	3
	7	Attente d'une action de l'utilisateur	1,25	
Dessins	8	Dessin d'un rectangle	2,00	4
	9	Mire	1,75	
Dessins à la souris	10	Mini Draw	4,75	5
			20,00	

Initiation à la programmation du simulateur

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

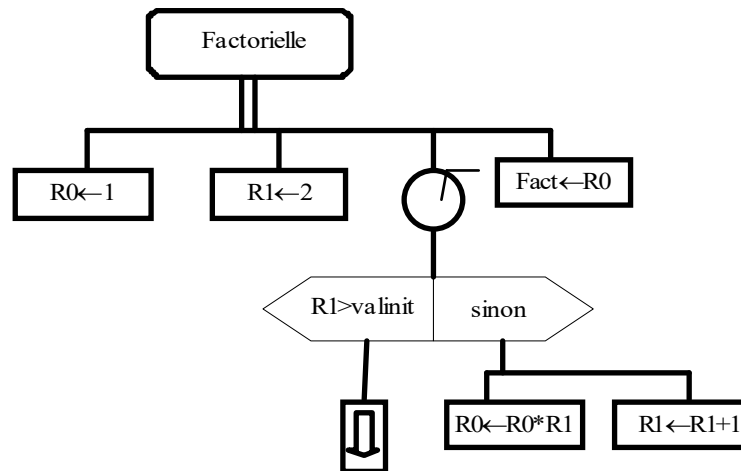
1. Apprendre le cours.
2. Lire le guide de programmation du simulateur du processeur en particulier le dernier paragraphe intitulé « Utiliser le simulateur ».
3. Concevoir un document texte manuscrit d'une page recto sur un papier A4 sans ligne pouvant être scanné automatiquement et contenant un résumé des informations essentielles nécessaires à vos débuts dans la programmation du simulateur.

Exercice 2: Tests des fichiers de démonstrations

1. A partir du cours d'elearn R2.04 Assembleur (<https://elearn.univ-pau.fr/course/view.php?id=18889>), copiez sur votre espace de travail le cours, le guide de programmation, le fichier de l'exécutable du simulateur de microcontrôleur, les fichiers de démonstration et ce sujet de TP. 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, réglez la vitesse d'horloge au maximum, décochez le mode pas-à-pas et exécutez le programme.
3. Rechargez le programme. Identifiez dans le simulateur l'affichage de la zone du microprocesseur. A partir de cette affichage, identifiez à quelle adresse mémoire se trouve la première instruction du code.
4. Visualisez cette première instruction dans la mémoire. En vous remémorant les codes écrits en cours et en TD, définissez le code écrit par le programmeur pour obtenir cette instruction en mémoire. Déduisez-en l'adresse de la pile et visualisez cette pile.
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 à l'aide du code couleur rouge pour l'écriture et vert pour la lecture.
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 124 du cours pour 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. Remarquez qu'un programme n'affiche pas forcément des informations à l'écran. Testez les programmes en vitesse normale. Pour le pong.mpc, utilisez les flèches pour déplacer la cible.

Exercice 3: Factoriel

L'objectif de cet exercice est d'écrire un programme en assembleur qui calcule la factorielle de 6. Ce programme est décrit par 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...).

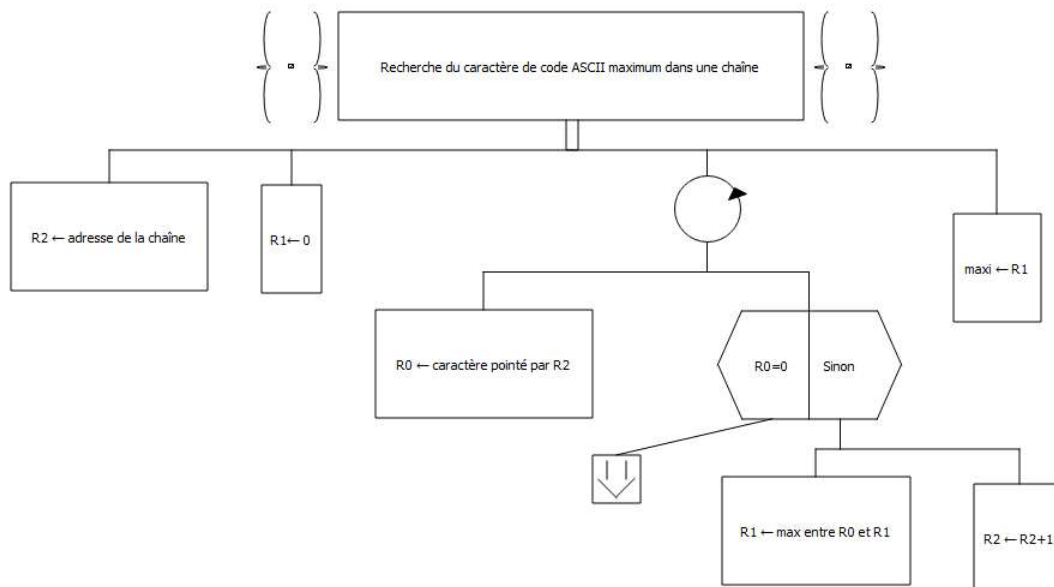
1. Etudiez l'algorithme afin d'en comprendre le fonctionnement.
2. Identifiez le test qui vous permettra de valider le programme réalisant cet algorithme.
3. Ecrivez le programme en assembleur. Si vous utilisez le Bloc Note, veillez à ce que l'extension soit correcte si besoin en décochant « masquer l'extension des fichiers connus » dans Options des dossiers de l'explorateur.
4. Lancez le simulateur et compilez ce programme. Lorsque la compilation est réussie, chargez le programme en mémoire. N'oubliez pas 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 exécutez le programme en pas à pas de façon à observer l'évolution du contenu de la variable 'Fact'. Vérifiez le résultat obtenu avec le test de la question 2.
5. Copiez le programme dans un nouveau fichier que vous renommerez puis modifiez ce nouveau programme pour que la variable 'valinit' soit initialisée à 13. Identifiez le nouveau test permettant de valider le programme. Exécutez le programme. Le résultat obtenu est-il correct ? Pourquoi ? Identifiez à quel moment et de quelle manière le simulateur vous indique ces informations en vous aidant de la description du registre d'état donnée en page 4 du guide de programmation et de la partie du cours sur ce même sujet.

Exercice 4: Recherche de caractères

Cet exercice a pour but de vous faire écrire un programme qui:

- déclare une chaîne de caractères contenant le texte de votre choix
- recherche dans cette chaîne le caractère ayant le code ASCII le plus élevé
- place celui-ci dans une variable appelée 'maxi'.

La chaîne de caractères sera terminée par 0 (la valeur 0 et non le caractère '0'). Le programme devra suivre l'algorithme ci-dessous :



1. Etudiez l'algorithme afin d'en comprendre le fonctionnement.
2. Identifiez le test qui vous permettra de valider le programme réalisant cet algorithme et précisez comment vous allez l'exécuter.
3. Ecrivez le programme.
4. Chargez ce programme en mémoire. Testez-le et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne. Quel est le caractère trouvé ?

Affichage de texte

Exercice 5: Soulignement de texte

On souhaite écrire un programme qui efface l'écran puis y affiche un texte. Le texte à afficher sera placé dans une variable initialisée de type chaîne de caractères terminée par un point. Le programme parcourt la chaîne caractère par caractère et affiche chaque caractère à l'écran jusqu'à rencontrer le point '.' qui ne sera pas affiché.

Notez que lorsque l'on écrit un caractère sur le périphérique du simulateur, les contenus des ports 1 et 2 sont automatiquement mis à jour pour pouvoir écrire le caractère suivant à l'endroit adéquat sans instruction supplémentaire que la commande d'écriture (voir guide de programmation page 6).

1. Identifiez le test qui vous permettra de valider le programme.
2. Ecrivez le programme puis testez-le et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.
3. Complétez le programme pour que le texte soit ensuite souligné : vous ne devrez pas modifier le code préalablement écrit mais juste y ajouter des instructions. Si le texte est modifié, le soulignement doit s'adapter automatiquement.

Vous noterez que quand on écrit un caractère à l'écran la coordonnée en x contenue dans Port 1 est mise à jour donc on peut savoir jusqu'où doit aller la ligne qui sert à souligner en relevant la valeur du Port 1 à la fin de l'écriture du texte. Le soulignement devra être visible à l'écran et donc suffisamment éloigné du texte pour ne pas le toucher, sans en être trop éloigné.

4. Identifiez le test qui vous permettra de valider le programme.

5. Testez le programme et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.

Utilisation des touches et de la souris

Exercice 6: Dessin conditionné par l'utilisateur

On souhaite écrire un programme qui :

- dessine un carré de 100 pixels de côté au centre de l'écran
- chaque fois que l'on appuie sur la touche A, le carré se réduit à 50 pixels de côté
- chaque fois que l'on lâche la touche A, le carré retrouve sa taille initiale
- lorsque l'on appuie sur la touche B, le programme se termine

Pour écrire ce programme vous définirez 2 procédures : l'une qui dessine un carré de 100 pixels de côté au centre de l'écran, l'autre qui efface l'écran puis y dessine un carré de 50 pixels de côté au centre de l'écran.

1. Identifiez le test qui vous permettra de valider le programme.
2. Ecrivez le programme
3. Testez le programme et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.

Exercice 7: Attente d'une action de l'utilisateur

1. Copiez le programme de l'exercice précédent et modifiez cette copie pour, qu'au début, il attende un clic de souris pour placer le carré de 100 pixels de côté centré sur le point où a eu lieu ce clic. Après quoi il fonctionne comme décrit à l'exercice précédent : réduction du carré, rétablissement et sortie du programme.
2. Identifiez le test qui vous permettra de valider le programme.
3. Testez le programme et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.

Dessins

Exercice 8: Dessin d'un rectangle

1. Ecrivez une procédure de dessin d'un rectangle plein qui reçoit dans la pile les 5 paramètres suivants et qui affiche le rectangle à l'écran.
 - un code de couleur (entier de 0 à 15)
 - la coordonnée en x du coin supérieur gauche du rectangle
 - la coordonnée en y du coin supérieur gauche du rectangle
 - la largeur du rectangle
 - la hauteur du rectangle
2. Ecrivez un programme permettant de tester la procédure.
3. Identifiez le test qui vous permettra de valider le programme.
4. Testez le programme et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.

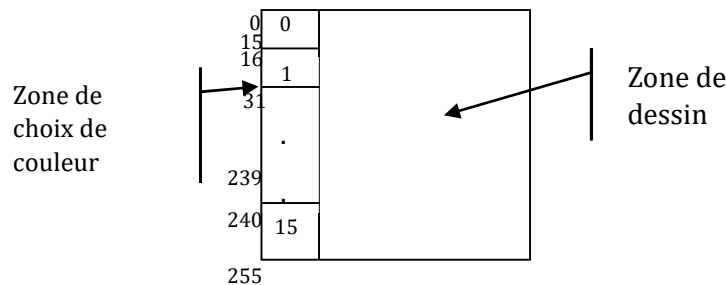
Exercice 9: Mire

1. Ecrivez un programme qui utilise cette procédure pour dessiner une mire. Cette mire sera constituée de 16 bandes horizontales de couleurs différentes occupant tout l'écran. Chaque bande aura donc une largeur de 255 pour une hauteur de 16.
2. Identifiez le test qui vous permettra de valider le programme.
3. Testez le programme et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.

Dessin à la souris

Exercice 10: Mini Draw

L'objectif est d'écrire un petit programme de dessin à la souris. L'écran se présente comme suit :

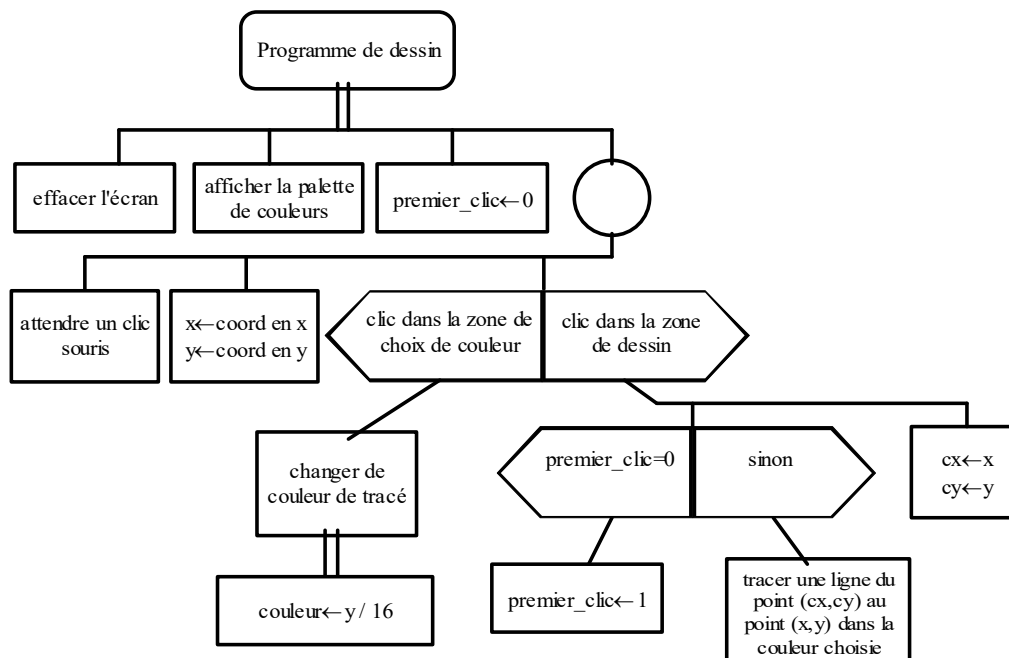


Dans la zone de choix de couleur on voit 16 rectangles de 20x16 chacun d'une couleur différente.

On cliquera sur l'un de ces rectangles pour choisir une couleur de tracé. Comme les rectangles représentant les couleurs ont une hauteur de 16 pixels, le numéro de la couleur correspondant à un rectangle peut être obtenu en divisant par 16 (division entière) la coordonnée en y de la souris lorsque l'on clique sur un choix (les couleurs sont codées entre 0 et 15).

Quand on clique dans la zone de dessin, une ligne est tracée entre le point où l'on vient de cliquer et le point précédent où l'on avait cliqué. Cette ligne est tracée dans la dernière couleur choisie. Le premier clic dans cette zone n'a donc aucun effet visible, la première ligne sera tracée entre ce point et le deuxième clic.

L'algorithme de ce programme est le suivant :



Remarque : pour savoir si un clic a eu lieu dans la zone de choix de couleur ou dans la zone de dessin, il suffit de regarder la coordonnée en x du clic : si elle est inférieure ou égale à 20 (largeur des rectangles représentant les couleurs) on est en zone de choix de couleur, sinon on est en zone de dessin.

1. Etudiez l'algorithme afin d'en comprendre le fonctionnement.
2. Identifiez le test qui vous permettra de valider le programme.

3. Réalisez le programme demandé.
4. Testez le programme et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.
5. Ajoutez au programme précédent la gestion des touches A et B de façon à ce que le bouton A permette d'effacer la zone de dessin et de réinitialiser premier_clic et le bouton B termine le programme.
6. Identifiez le test qui vous permettra de valider le programme.
7. Testez le programme et, si besoin, corrigez-le jusqu'à ce qu'il fonctionne.