Université de Liège

Systèmes Programmés Enfouis : Laboratoire 3

Hugues Smeets

Année académique 2001-2002

1 L'objectif du travail

Ce travail a pour but de vous entraîner à programmer des traitements relativement complexes en assembleur pour PIC 16f877. L'architecture RISC de ce microcontrôleur débouche sur un jeu d'instructions particulier par rapport aux processeurs plus classiques (Z80, 68xx, 80x86, ...). La transcription, dans cet assembleur, d'un algorithme peut être non triviale, même si vous disposez de toutes les informations nécessaires (analyse précise du problème, algorithme sous forme de pseudo-code, ...).

Je vous propose de réaliser, en fonction du temps disponible, trois programmes...

2 Retournement de matrice

Cette technique a très souvent été employée au cours de l'histoire de l'informatique. En effet, imaginez le problème posé par une imprimante matricielle à huit aiguilles. La tête d'impression est constituée de huit aiguilles disposées verticalement et se déplace horizontalement. Le papier est déplacé par un rouleau. Si vous voulez afficher un caractère dont l'aspect est codé sous forme d'une matrice de huit lignes de huit colonnes, et qu'un octet représente une ligne, vous devrez envoyer à la tête d'impression les *colonnes* de la matrice en la retournant de 90 degrés (on pourait aussi coder directement les colonnes sous forme d'octets mais le listing serait moins lisible et ce serait moins amusant !).

Un autre exemple est celui d'une carte graphique où les pixels peuvent prendre 256 valeurs possibles et dont la mémoire serait organisée sous forme de huit plans monochromes. Huit pixels consécutifs sur une même ligne posséderaient chacun un bit dans chacun des huit plans de mémoire. Si vous disposez des huit couleurs de huit pixels consécutifs, vous devrez retourner cette matrice de 90 degrés pour obtenir les huit octets à stocker dans les huit plans de mémoire. Cet exemple est illustré sur la figure 1.

Beaucoup d'anciennes cartes graphiques utilisaient cette technique car elle permet d'utiliser huit circuits de mémoire qui peuvent être huit fois plus lents qu'un circuit de mémoire unique (et qui contiendrait directement un octet par pixel).

Je vous suggère donc d'écrire un programme, à tester avec le simulateur, qui opère cette rotation de 90 degrés sur une matrice de huit octets se trouvant dans la RAM du PIC.

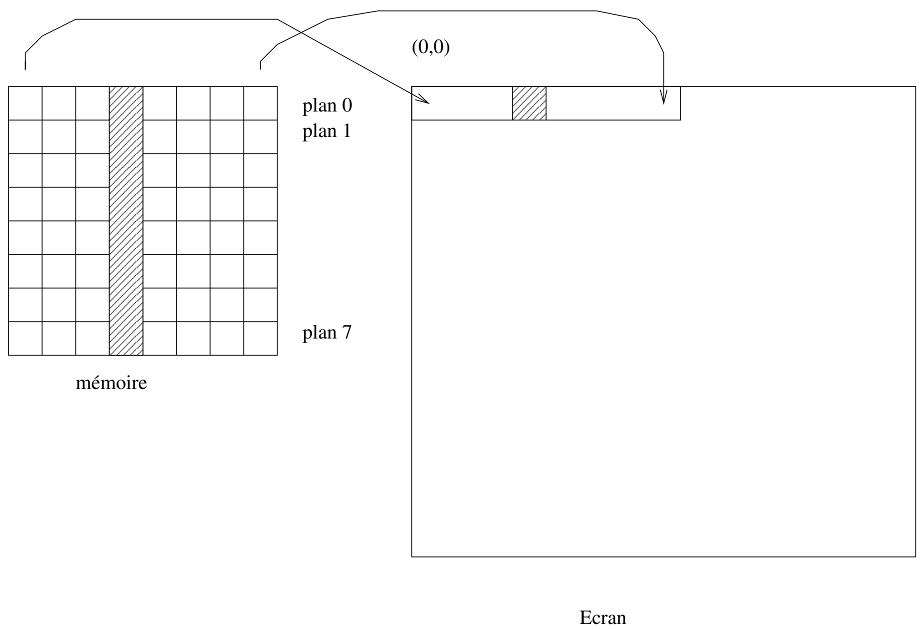

FIG. 1 – 8 bits par points et 8 plans mémoire

3 Implémentation d'un tri rapide sans pile

Après l'exercice précédent, qui a servi d'échauffement, passons au plat de résistance !

Le PIC ne possède pas de pile matérielle permettant de stocker des données. Pour implémenter un tri de type "quick sort", on pourrait travailler avec une pile *software*, mais une solution plus élégante existe : le "one way, stackless quick sort"[1], décrit en détail dans le cours d"'Algorithmique 3" donné par P.A. de Marneffe. Le PIC est le terrain idéal pour mettre en œuvre cet algorithme. Voici une spécification de cet algorithme en méta-langage (l'invariant correspondant est donné en figure 2) :

```
A[N] := -1; l := 0; nb := 0

;do l < N ->
    if A[l] < 0 -> A[l] := -A[l]; l := l + 1
    [] A[l] > 0 -> ``placer A[l]'' ; nb := nb + 1
    fi
od
```

[1]Huang Bing-Chao, Donald E. Knuth

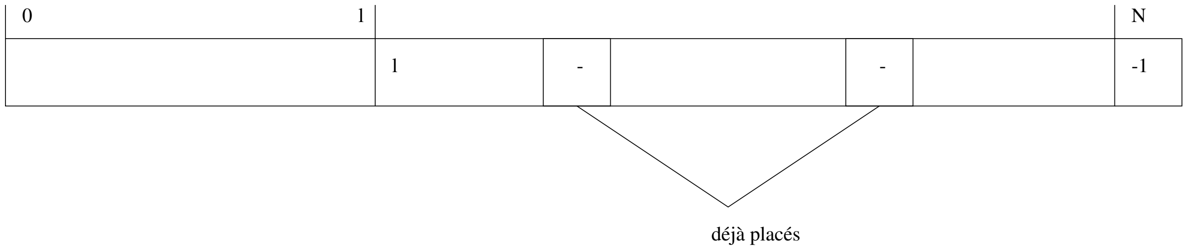

FIG. 2 – Invariant principal

la fonction t est : $t = 3N - nb - 2l$. La variable nb est inutile, elle n'est présente que pour spécifier la fonction t.

L'algorithme "placer A[l]" est le suivant (voir l'invariant en figure 3) :

```
p := A[l]; i,j := l,l+1

;do A[j] > 0 ->
    if A[j] >= p -> j := j + 1
    [] A[j] < p -> A[i] := A[j] ; i := i + 1
                  ;A[j] := A[i] ; j := j + 1
    fi
od
;A[i] := -p
```

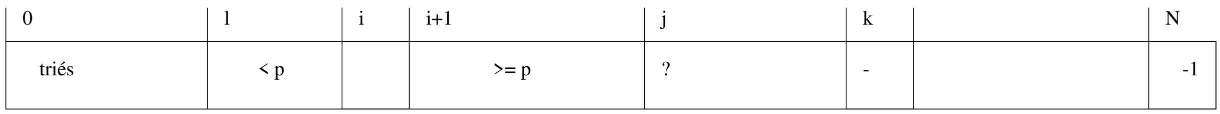

FIG. 3 – L'invariant de "placer A[l]"

La fonction t vaut ici $t = k - j$. Remarquez que les clés à trier doivent être strictement positives. Je vous propose d'implémenter cet algorithme en assembleur pour le PIC 16f877. Bon travail !

Si vous avez terminé ce deuxième exercice et qu'il vous reste du temps, tentez d'écrire une procédure qui extrait la racine carrée d'un nombre 16 bits (sans consulter les exemples fournis par Microchip)...