T.D. 12 – Corrigé
Exercices de programmation 68000

Exercice 1

Réalisez un sous-programme récursif **facto** qui calcule la factorielle d'un nombre entier **n** (aucun registre hormis **D0** ne sera modifié en sortie du sous-programme).

Entrée : **D0.W** = **n** (entier non signé codé sur 16 bits)

Sortie : **D0.L** = **n!**

```
facto               tst.w   d0          ; Si n = 0, saut à quit
                    beq.s   quit

                    move.w  d0,-(a7)    ; Sauvegarde n dans la pile
                    subq.w  #1,d0       ; n - 1     → d0
                    bsr.s   facto       ; (n - 1)!  → d0
                    mulu.w  (a7)+,d0    ; n(n - 1)! → d0 (n est dépilé)

                    rts                 ; Sortie

quit                moveq.l #1,d0       ; 0! → d0
                    rts                 ; Sortie
```

Exercice 2

Soit un mot de 16 bits : $\mathbf{X_3\ X_2\ X_1\ X_0}$ (X_n représentant un paquet de quatre bits). Écrivez un programme, en assembleur 68000, qui inverse les paquets de chaque octet, c'est-à-dire qui a pour résultat le mot suivant : $\mathbf{X_2\ X_3\ X_0\ X_1}$. On supposera qu'une valeur de départ est contenue dans le registre **D1**. Vous avez à votre disposition tous les autres registres du 68000.

```
                                ; d1 = X3 X2 X1 X0
                    ror.b   #4,d1   ; d1 = X3 X2 X0 X1
                    ror.w   #8,d1   ; d1 = X0 X1 X3 X2
                    ror.b   #4,d1   ; d1 = X0 X1 X2 X3
                    ror.w   #8,d1   ; d1 = X2 X3 X0 X1
```

Exercice 3

Réalisez une multiplication par 2 d'un nombre 128 bits en respectant les indications suivantes :

Entrées : **D3:D2:D1:D0** = Entier non signé codé sur 128 bits (**D0** étant les 32 bits de poids faible).

Sortie : **D3:D2:D1:D0** = **D3:D2:D1:D0** × 2

La multiplication par 2 peut être obtenue à l'aide d'un décalage d'un bit vers la gauche des 128 bits. Les instructions de rotation et de décalage sont limitées à une taille de 32 bits. Il faut donc réaliser 4 décalages vers la gauche. À chaque décalage, le bit sortant doit être mémorisé et réinjecté dans les 32 bits suivants. C'est le *flag* **X** qui sera utilisé à cet effet.

On obtient le programme suivant :

```
        lsl.l   #1,d0     ; X ← [d0] ← 0
        roxl.l  #1,d1     ; X ← [d1] ← X
        roxl.l  #1,d2     ; X ← [d2] ← X
        roxl.l  #1,d3     ; X ← [d3] ← X
```

Exercice 4

Proposez quelques lignes en assembleur 68000 qui échangent le contenu des registres **D0.B** et **D1.B** sans passer ni par un troisième registre, ni par la pile, ni par la mémoire.

```
                    ; Soit d0 = a et d1 = b
                    ; on sait que :
                    ; Si    c = a ⊕ b
                    ; alors a = b ⊕ c
                    ; et    b = a ⊕ c
                    ; Ce qui donne :
                    eor.b   d0,d1       ; a ⊕ b = c → d1
                    eor.b   d1,d0       ; c ⊕ a = b → d0
                    eor.b   d0,d1       ; b ⊕ c = a → d1
```