Architecture des Ordinateurs et Microprocesseurs

Exercices récapitulatifs

Partie E

Exercice 6 :

1) Parmi ces instructions 68000, trois sont incorrectes. Il vous est demandé de les trouver (cochez la case correspondante) en précisant la source d'erreur.

MOVE.B D1,1(A0)	.....	☐
MOVE.W #$D0,D0	.....	☐
ROXL.B #$01,D8	.....	☑
LEA $1003,A0	.....	☐
ADDQ.W #$1A,D5	.....	☑
MOVE.W $102B,D3	.....	☐

2) Donnez la taille (nb d'octets) de la représentation-mémoire de ces instructions

MOVE.W $80004,D3 ..... *octets*

MOVEQ $04,D0 ..... *octets*

MOVE.L (A0)+,D0 ..... *octets*

Exercice 7 :

Pour le calcul des lois de pilotage d'un avion, au sein des ELAC (Elevator and Aileron Computer), on se propose d'utiliser le microprocesseur 68000 de Motorola.

Ces calculs imposent des méthodes d'optimisations non conventionnelles basées sur les algorithmes évolutionnaires travaillant sur un ensemble de solutions (individus) en parallèle appelé population qui évolue de génération en génération à l'aide d'opérateurs de sélection, de croisement et de mutation.

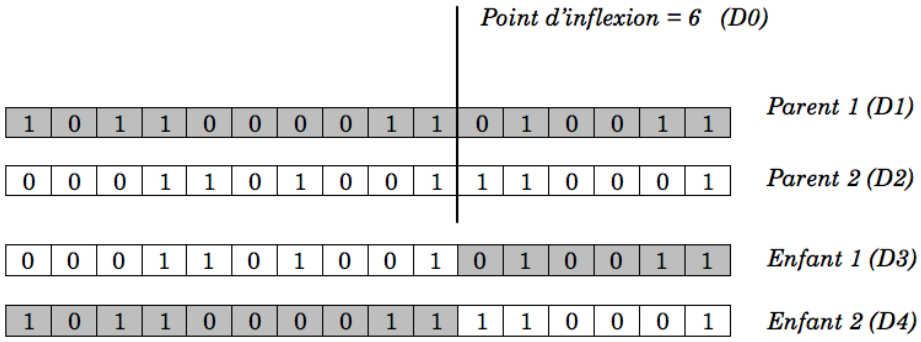
On se limitera, dans cet exercice à l'étude de la technique de croisement.

Proposez un sous-programme assembleur 68000 permettant de réaliser le croisement de deux individus (parents) codés en binaire respectivement dans les 16 premiers bits de D0 et D1 et de stoker les deux individus ainsi obtenus (enfants) respectivement dans D3 et D4 sachant que le point d'inflexion utilisé doit être lu à partir du registre D2.

Architecture des Ordinateurs et Microprocesseurs

Exercices récapitulatifs

Exemple :



Indication : Vous pouvez utiliser les deux masques déclarés comme suit :

Masque1 DC.W $0000, $0001,$0003,$0007,$000F,$001F,$003F,$007F
DC.W $00FF, $01FF,$03FF,$07FF,$0FFF,$1FFF,$3FFF,$7FFF
Masque2 DC.W $FFFF,$FFFE,$FFFC,$FFF8,$FFF0,$FFE0,$FFC0,$FF80
DC.W $FF00, $FE00,$FC00,$F800,$F000,$E000,$C000,$8000

Rqs :
- Vous pouvez utiliser votre propre méthode.
- Si vous utilisez l'indication précédente, vous n'êtes pas appelé à la réécrire.

Exercice 8 :

On dispose en mémoire d'un texte, c'est à dire d'une suite de codes ASCII. On désire envoyer ce texte par un réseau informatique après l'avoir transformé pour qu'il ne puisse pas être lu facilement. Pour cela on va le chiffrer par une méthode simple: pour chaque octet du texte, on va permuter les groupes de bits 0,1,2 et 4,5,6 (les bits 0 et 4 seront échangés, de même que les bits 1 et 5, et ainsi de suite).

1) Proposez un sous-programme nommé *"permute"* permettant de permuter les bits 0,1,2 avec les bits 4,5,6 du registre D0.

2) Proposez un sous-programme nommé "modify" permettant de modifier ainsi une suite d'octets à partir de l'adresse indiquée par le registre A1 et pour un nombre d'octets indiqué dans le registre D1.