a) Variables et constantes :
---

l10 : Cette ligne sert à inclure un fichier contenant diverses définitions de constantes, variables et fonctions diverses.

l123,124 : Les variables 'OptosLGN' et 'memOptosLGN' varient sur l'intervalle [0, 255] (unsigned char)

l136,140 : la variable 'clk.sec' est définie sur l'intervalle [0, 65535] (unsigned int)

l144,146 : ces trois lignes définissent les constantes SET, RESET et CLEAR valant respectivement 1, 0 et 0.

l159 : la variable T1CON est définie sur l'intervalle [0, 255]

b) Programme principal :
---

le programme principal est défini de la ligne 13 à la ligne 56

l15 : à la ligne 15 ,la fonction Init() est appelée. (fonction = sous-programme)

l17 : les variables 'TMR0IF' et 'RBIF' valent 0 (CLEAR) après l'exécution de cette ligne.

l20,23 : Ces 4 lignes permettent d'attendre que la variable 'clk.sec' prenne la valeur '1'
(temporisation de 1 seconde après la mise sous tension)

l29 : cette ligne donne la valeur binaire '00010001' à la variable T1CON.
Cette valeur de 8 bits est bien compatible avec la définition de la variable (ligne 159).
On aurait pu ecrire la ligne sous la forme T1CON = 0x11; ou T1CON= 17;

l34 : cette ligne effectue la division entière de la constante TRANCHE_MOT par 4 puis range le résultat dans les deux variables 'mot.gauche' et 'mot.droit'.
La constante TRANCHE_MOT est définie en ligne 143 avec la valeur 0x10 = 16

l36,55 : la ligne 151 définit FOREVER = '1'. La valeur '1' étant différente de zéro , elle correspond dans un test à une con toujours VRAIE. On ne sortira donc JAMAIS de la boucle en question.

l43 : cette ligne appelle la fonction Gere_Ligne() La prochaine instruction exécutée se trouve en ligne 66.

l66,72 : Il faut que les deux variables 'OptosLGN' et 'memOptosLGN' soient différentes pour que soit exécuté le bloc d'instructions [l68..l71] Si cette condition n'est pas remplie on retourne dans le programme principal en ligne 45

l45,53 : si la variable 'stop' vaut '0' (0 = FAUX) on exécute la partie ELSE du test : lignes 51 et 52 .
si cette variable vaut '1' (1= VRAI) on éxécute la ligne 47.

c) Sous-Programme (fonction) ReadLigne() :
---

l87 : les variables 'li', 'lj' et 'lk' sont définies sur l'intervalle [-128, 127]
les variables li, et lk recoivent les valeurs initiales 0
la variable lj reçoit la valeur 255 (0xFF). Cette valeur est donc hors intervalle de définition

A priori il s'agit d'une erreur du programmeur !!!

(Remarque :Le compilateur HiTech utilisé possède en fait une option grâce à laquelle tous les 'char' sont 'unsigned'.....le problême est donc évité....)

Il faut que la variable 'l' soit inférieure à '7' pour que soit répétée l'exécution de la boucle [l89..l100]

l'expression 'li++' équivaut à 'li = li + 1;' (= incrémentation).

l89 : compléméntation de PORTA, suivie d'un quadruple décalage à droite.
Le résultat subit ensuite un ET bit à bit avec le nombre 7

exemple :

si PORTA vaut 01110000, la complémentation donnera le nombre 10001111
après le quadruple décalage on aura le nombre 00001000
)ET = masque
le nombre 7 , ecrit sur 8 bits est 00000111
le résultat final sera donc ==> 00000000

l91 : Pour exécuter la ligne 98 il faut que la parenthèse de la ligne soie FAUSSE (lj différent de OptoslGN)

(lj == OptosLGN) signifie 'lj' est-il égal à OptosLGN ? c'est un test, une question.
(lj = OptosLGN) signifie 'lj'prend la valeur actuelle de OptosLGN.

l107 : Cette ligne ne sera exécutée que si OptosLGN vaut zéro ET si simultanément memOptosLGN possède l'une des trois valeurs suivantes : 1, 3 ou 8