Notes d'applications
Volume #B

"Everything for Embedded Control"

COMFILE
TECHNOLOGY

Copyrights et appellations commerciales

Toutes les marques, les procédés et les références des produits cités dans ce document appartiennent à leur propriétaire et Fabricant respectif. All brand names and trademarks are the property of their respective owners - Other trademarks mentioned are registered trademarks of their respective holders.

Informations techniques

Les notes d'applications décrites dans ce document ont été conçues avec la plus grande attention. Tous les efforts ont été mis en oeuvre pour éviter les anomalies. Toutefois, nous ne pouvons garantir que ce document soit à 100% exempt de toute erreur. Les informations présentes dans ces notes d'applications sont strictement données à titre indicatif. Les caractéristiques et les résultats obtenus par ces notes d'applications peuvent changer à tout moment sans aucun préavis.

Limitation de responsabilité

En aucun cas LEXTRONIC ne pourra être tenu responsable de dommages quels qu'ils soient (intégrant, mais sans limitation, les dommages pour perte de bénéfice commercial, interruption d'exploitation commerciale, perte d'informations et de données à caractère commercial ou de toute autre perte financière) provenant de l'utilisation ou de l'incapacité à pouvoir utiliser les notes d'applications décrites dans ce document, même si LEXTRONIC a été informé de la possibilité de tels dommages.

Ces notes d'applications sont uniquement destinées à être utilisées telles quelles dans le cadre d'un apprentissage à la programmation des modules "CUBLOC™". LEXTRONIC ne donne aucune garantie de fonctionnement de ces notes d'applications si vous utilisez celles-ci au sein d'une autre application. A ce titre, ces notes d'applications ne sont pas conçues, ni destinées, ni autorisées pour être utilisées au sein d'applications militaires, ni au sein d'applications à caractère médical ou d'alerte incendie, ni au sein d'applications pour ascenseurs ou commande de feux d'artifices, ni au sein d'applications sur machine outils ou d'applications embarquées dans des véhicules (automobiles, camions, bateaux, scooters, motos, kart, scooters des mers, avions, hélicoptères, ULM, etc...), ni au sein d'applications embarquées sur des maquettes volantes de modèles réduits (type avions, hélicoptères, planeurs, etc...).

De manière générale, ces notes d'applications ne sont pas conçues, ni destinées, ni autorisées pour expérimenter, développer ou être intégrées au sein d'applications dans lesquelles une défaillance des modules "CUBLOC™" pourrait créer une situation dangereuse pouvant entraîner des pertes financières, des dégâts matériels, des blessures corporelles ou la mort de personnes ou d'animaux. Si vous utilisez ces notes d'applications associées aux modules "CUBLOC™" ainsi que leurs platines et modules optionnels associés volontairement ou involontairement pour de telles applications non autorisées, vous vous engagez à soustraire le Fabricant et LEXTRONIC de toute responsabilité et de toute demande de dédommagement.

L'exploitation de ces notes d'applications nécessite que l'utilisateur respecte également toutes les précautions d'utilisations relatives à la mise en œuvre des modules "CUBLOC™" (lesquelles sont détaillées dans la documentation de ces derniers).

Liste des notes d'applications par ordre chronologique de parution

Liste par thèmes"

NOTE D'APPLICATION # 31.
Gestion d'un module ultrason « MS-EZ1 »

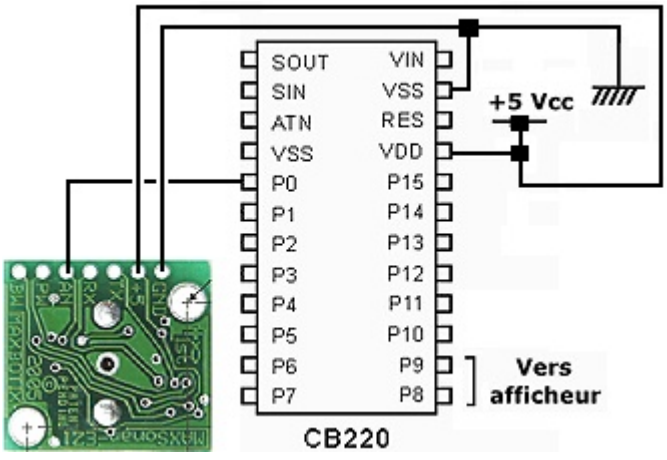
Cette note d'application va vous permettre de piloter un module ultrason miniature « MS-EZ1 » capable de mesurer la distance qui le sépare d'un obstacle. Ce module est équipé d'une unique cellule ultrason « émetteur/récepteur » 42 KHz associée à un microcontrôleur et à divers composants électroniques. Ce dernier est capable de détecter la présence d'un objet entre 0 et 6,45 mètres et de vous donner une indication sur la distance qui le sépare du module lorsque cet obstacle est compris entre 15,24 cm et 6,45 mètres avec une résolution de l'ordre de 2,54 cm env. Les objets distants de 0 à 15,24 cm retournent une information de distance comme si ils étaient à 15,24 cm.

L'information de distance peut être récupérée via 3 signaux différents : Une valeur analogique comprise entre 0 et 2,55 V (proportionnelle à la portée), un signal PWM dont la largeur d'impulsion est également proportionnelle à la portée ainsi qu'un signal série (0 – 5V) qui fourni une information directe de la portée (en inch). Nous utiliserons la sortie analogique puis la sortie série pour les besoins de l'application

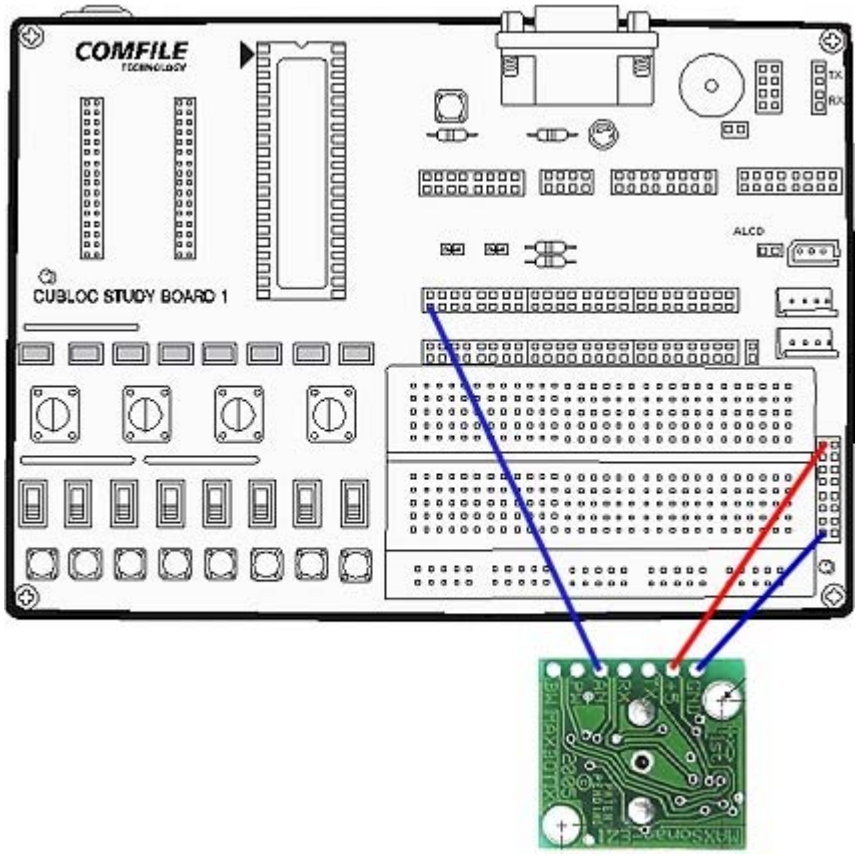
Notions abordées :
- Conversion « analogique/numérique »
- Gestion communication série

Matériel nécessaire :
- Une platine « CUBLOC Study Board » (facultative)
- Module « MS-EZ1 » (+ 2 résistances 10 Kohms + 1 transistor BC338)

Préparation matérielle (utilisation du module via la sortie analogique):
Celle-ci repose sur le schéma théorique ci-dessous.



Afin de faciliter la description de cette note d'application, nous utiliserons une platine « CUBLOC Study Board » associée à un module CUBLOC™ CB220 (voir schéma de raccordement ci-après).



Saisissez ensuite le petit programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « MaxSonar1 »).

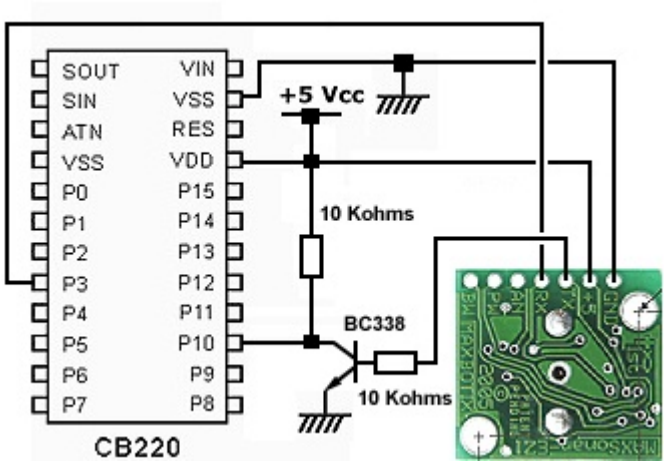
L'application consistera à récupérer la valeur analogique du module « MS-EZ1 » et d'en afficher une représentation au moyen d'un bargraph « haute résolution ». Ce bargraph a déjà fait l'objet d'une description dans la note d'application #1 (voir fichier PDF des notes d'applications #A). Ainsi en déplaçant un objet devant le module « MS-EZ1 », vous obtiendrez un déplacement proportionnel du bargraph.

La valeur de la tension mesurée est multipliée par 1,96 car la valeur max. en sortie du module « MS-EZ1 » est de 2,55 V. Afin de pouvoir bénéficier de pla plaine échelle du bargraph qui est pour + 5 V, on introduit n facture multiplicateur de 1,96 (1,96 x 2,55 = 5).

```
'###################################
'#   Gestion d'un module « MS-EZ1 »      #
'# @Lextronic 2006 - 03/07/2006          #
'###################################

  Const Device = CB220                    ' Initialisation de l'afficheur LCD
  Set Display 2,0,1,50
  Dim AN As Integer

  Delay 100
  Cls
  Delay 200
  Csroff
  Delay 100

  Print &H1B,&H44,8,0,0,0,0,0,0,0 ' Redéfinition des caractères du LCD
  Print &H1B,&H44,9,16,16,16,16,16,16,16
  Print &H1B,&H44,10,24,24,24,24,24,24,24
  Print &H1B,&H44,11,28,28,28,28,28,28,28
  Print &H1B,&H44,12,30,30,30,30,30,30,30

  Input 0                                 ' Configure les ports de conversion analogique/numérique en entrée

  Do
     AN = Adin(0)                         ' Lecture de la tension analogique du module MaxSonar
     AN = AN*1.96
     bargraph 0                           ' Affichage de la valeur sous forme de bargraph sur la 1ère ligne
   Loop
   End


'############################
'# Sous Routine BARGRAPH  #
'############################

  Sub bargraph(ligne As Byte)
    Dim i As Byte
    Dim J As Byte
    Dim valmax As Single
    Dim barre As String*20
    Dim vide As String*20
    Dim segment As String*1
    valmax=(AN/10.23)/5
    i=valmax
    barre=String(&HFF,i)
    J=20-i
    vide=Spc(J)
    i=((valmax-i)/0.2)+8
    segment=Chr(i)
    barre=barre+segment+vide
    Locate 0,ligne
    Print barre
  End Sub
```

Préparation matérielle (utilisation du module via la sortie série):

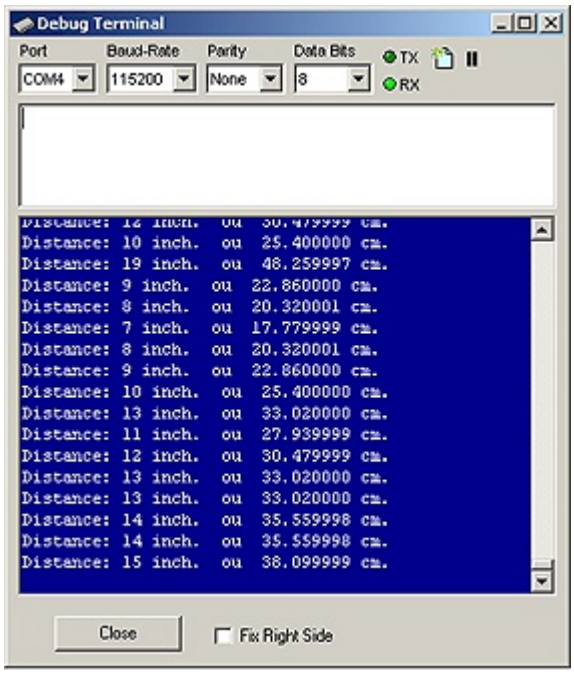
Pour la 2ème partie de la note d'application on va interfacer le module avec son signal série. Le signal étant inversé par rapport au standard RS-232, on utilise un simple étage à transistor pour le remettre en forme. Réalisez alors le schéma théorique ci-dessous.



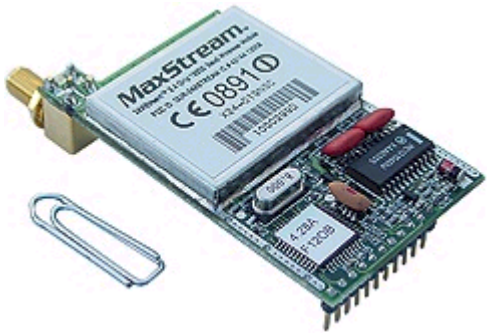
Saisissez ensuite le petit programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « MaxSonar2 »).

```
'#################################
'#   Gestion d'un module « MS-EZ1 »      #
'# @Lextronic 2006 - 03/07/2006          #
'#################################

  Const Device = CB220

  Dim cm1 As Byte
  Dim cm2 As Single
  Dim inch As String*5

  Opencom 1,9600,3,30,20                  ' Initialisation pour série
  Low 3                                                ' Configure port en sortie et stop mesure distance
  Delay 500

  Do
     Bclr 1,0                             ' Efface buffer réception
     High 3                               ' Lance la mesure de distance
     Do While Blen (1,0) = 0              ' Attend la réception du module "MaxSonar"
     Loop
     Low 3                                ' Stop la mesure
     Delay 100
     inch = Getstr(1,4)
     cm1 = Val(Right(inch,3))
     Debug "Distance: ",Dec cm1," inch.  ou  "
     cm2 = cm1 *2.54
     Debug Float cm2," cm.",Cr
  Loop
```

Le programme est très simple à assimiler. Le port P3 est utilisé pour activer une mesure de distance en passant du niveau bas au niveau haut. Le CUBLOC attend alors de recevoir au moins un octet de la part du module « MS-EZ1 », puis après une petite temporisation, il va récupérer les 4 premiers octets reçus. Parmi les 4 octets, le premier est toujours le caractère ASCII « R ». Le programme va donc convertir les valeurs ASCII des 3 autres caractères (qui correspondent à la portée exprimée en inch). Cette valeur est alors convertie en cm par une simple multiplication par 2,54 (1 inch = environ à 2,54 cm). Dès lors la fenêtre DEBUG du PC va afficher à la fois la distance mesurée en inch et en cm.



NOTE D'APPLICATION # 32.
Gestion d'un modem radio FHSS « X24-009 »



Cette note d'application va vous permettre de piloter un Module radio FHSS « X24-009 » du fabricant Maxstream™. Ce dernier est un modem basé sur une technologie d'étalement de spectre à saut de fréquence (Frequency Hopping Spread Spectrum). Il intègre une "électronique intelligente" qui vous permettra de réaliser aussi bien des communications "basiques" point-à-point que de véritables réseaux de transmission de données multi-points. Il assurera la transmission à distance de signaux numériques séries de façon totalement transparente (le module génère les trames de préambule, de synchro, ainsi que la mise en "paquet" et le codage des données tout en effectuant un checksum). D'un point de vue utilisation celui ci s'apparente en fait à un simple câble série « virtuel » très rapide et simple à mettre en œuvre.

Notions abordées :
- Gestion communications séries.

Matériel nécessaire :
- Une platine « CUBLOC Study Board » (facultative)
- 2 modules « X24-009 » (ou un module « X24-009» et un module « X24-009DK »

Cette note d'application vous montera comment interfacer ce modem avec un CB220 afin de pouvoir réaliser une télécommande radio 4 canaux bidirectionnelle (via un compatible PC) avec fonction d'accusé de réception des ordres. Le module « X24-009 » dispose d'une interface série (niveau logique 0 / + 5V qui pourra très facilement s'interfacer directement avec le CUBLOC). Du côté du PC, il vous faudra ajouter une circuit de mise à niveau MAX-232 afin d'adapter les signaux +/-12 V du port RS-232 avec les niveaux du « X24-009 » (vous trouverez un schéma d'application « type » sur le site du fabricant : www.maxstream.net).

Il est également possible d'avoir recours à un module en boîtier « X24-009DK », lequel intègre sur une même platine le modem radio, une prise SUB-D 9 broches, son antenne et le circuit d'interfaçage Max-232 (de telle sorte qu'il vous suffira simplement de le relier au port RS-232 du PC pour le rendre opérationnel).

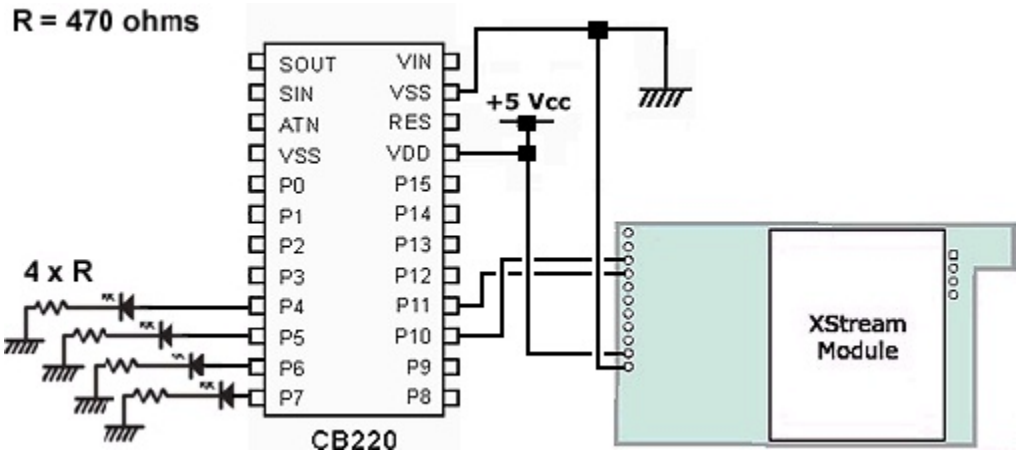
Préparation matérielle :

Du côté du PC, celle-ci consiste à relier le boîtier « X24-009DK » à un port COM de libre du PC et à exécuter le programme de communication « HyperTerminal™ ». Ce programme se trouve via les menus :

Démarrer -> **Programmes** -> **Accesoires** -> **Communications** -> **HyperTerminal.**

Une fois le programme exécuté, donnez un nom à la communication (par exemple CUBLOC), puis sélectionnez le N° de port sur lequel est connecté le module « X24-009DK » dans l'onglet du bas de la fenêtre. Validez par OK. Sélectionnez enfin une vitesse de connexion de 9600 bds / 8 bits / Parité (aucun) / 1 bit de stop / Contrôle de flux (aucun) et validez à nouveau. Cliquez ensuite sur l'onglet « Paramètres », puis cliquez sur « Configuratio ASCII… » et validez la sélection « Ajouter les changements de ligne à la fin des lignes entrantes ». Alimentez alors le module « X24-009DK » à l'aide d'un bloc d'alimentation secteur 220 VAC/12Vcc.

Du côté du module CUBLOC, il vous faudra réaliser le montage ci-dessous.



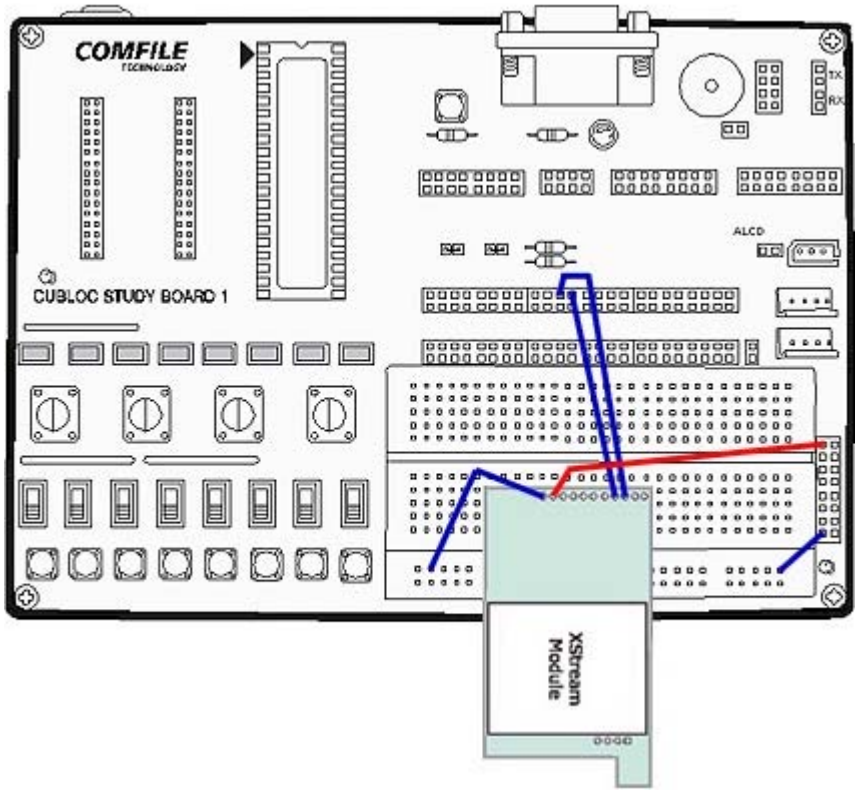
Principe de fonctionnement de l'application :

Le programme HyperTerminal™ présent sur le PC vous permettra d'envoyer des ordres séries depuis le clavier de votre PC, lesquels seront transmis au module CUBLOC via les 2 modems radio. Ainsi en sollicitant les touches 1 à 4 du PC, vous changerez alternativement l'état des 4 sorties du CUBLOC (une sollicitation de la touche 1 du PC allume la Led de la première sortie du CUBLOC – une seconde sollicitation de la touche 1 du PC éteint la Led de la première sortie et ainsi de suite pour les touches 2 à 4 et les sorties Leds 2 à 4 du module CUBLOC).

Le programme HyperTerminal™ peut également recevoir des données en provenance du port sortie en les affichant à l'écran. La note d'application utilise cette possibilité pour renvoyer après chaque commande l'état des 4 sorties du CUBLOC. Ainsi après avoir sollicité une des touches 1 à 4 au niveau du PC, le CUBLOC change l'état de la sortie adéquat et renvoie sous forme de textes l'état des 4 sorties qui s'affichera sur la fenêtre du PC (vous serez ainsi sûr que l'ordre radio a correctement été reçu et que la sortie a bien l'état que vous vouliez lui faire prendre).

En sollicitant la touche 0 du PC, le CUBLOC ne modifie aucun état de ses sorties, mais renvoie tout de même l'état de ces dernières. La touche 0 fait ainsi office de touche d'interrogation en quelque sorte (dans le cas de figure où vous avez un doute sur l'état des sorties du module CUBLOC).

Réalisation du montage à l'aide de la platine « Cubloc Study Board » :



Afin de faciliter la mise en œuvre du modem radio côté CUBLOC, il vous sera possible d'utiliser une platine «CUBLOC Study Board» comme le montre le schéma ci-dessus.

Saisissez ensuite le petit programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « X24-009 »).

Interprétation du programme :
Le programme commence par l'initialisation des variables représentant l'état des 4 sorties du module CUBLOC (et de leur initialisation « physique » au niveau logique 0 V). Le programme continu ensuite par l'analyse du contenu du buffer série de réception du module CUBLOC. Si le CUBLOC a reçu un code ASCII correspondant à une des touches 1 à 4 du PC, il s'en suit un changement d'état intermittent (via une fonction XOR) des sorties en fonction du caractère reçu dans le buffer. La valeur 49 correspond à la valeur ASCII du chiffre de la touche 1 sollicitée sur le PC. La valeur 50 correspond au chiffre 2, etc…Une fois la variable de la sortie remise à jour, le programme force la variable de réception avec la valeur 48 (correspondant à la valeur ASCII du chiffre de la touche 0 du clavier – ceci afin de « forcer » le reste du programme à renvoyer un accusé de réception). Le programme remet ensuite à jour l'état « physique » des sorties puis envoi un accusé de réception de l'état de chaque sortie via des phrases dont chaque lettre est transmise au module radio « X24-009 » afin que ces phrases s'affichent dans la fenêtre de HyperTerminal™ du PC.

NOTE IMPORTANTE :
Il est important de signaler que le programme de cette note d'application ne fait office que de programme d'évaluation et de test et qu'il ne devra en aucun cas être utilisé tel quel au sein d'une application. En effet, les sorties du CUBLOC sont directement activées via la réception d'un seul octet. Dès lors n'importe quel autre module « X24-009 » émettant à proximité de votre application pourra être susceptible de déclencher vos sorties si ce dernier envoi un code ASCII correspondant aux touches 1 à 4. Dans le cadre d'une application réelle, il vous faudra impérativement envoyer au préalable un code sur plusieurs octets correspondant à une sorte de « mot » de passe ou de « clef de sécurité » (ce code pourra être répété avant et après les ordres de commande). Dès lors le CUBLOC devra avant toute activation de ses sorties s'assurer de la validité de ce code afin que vous soyez sûr qu'il s'agisse bien de votre application qui essai de vous envoyer un ordre radio.

```
'####################################
'#        Gestion d'un modem radio          #
'#       Maxstream™ "X24-009"               #
'#     @Lextronic 2006 - 05/07/2006         #
'####################################

Const Device = CB220

Dim sortie1 As Byte
Dim sortie2 As Byte
Dim sortie3 As Byte
Dim sortie4 As Byte
Dim reception As Byte

Opencom 1,9600,3,10,35

'****************** RAZ sorties *****************
Low 4
Low 5
```

```
Low 6
Low 7
Sortie1 = 0
Sortie2 = 0
Sortie3 = 0
Sortie4 = 0

'****************** Boucle principale *************
Do
   reception=Get(1,1)
   Select Case reception
     Case 49
       sortie1=sortie1 Xor &HFF
       reception=48
     Case 50
       sortie2=sortie2 Xor &HFF
       reception=48
     Case 51
       sortie3=sortie3 Xor &HFF
       reception=48
     Case 52
       sortie4=sortie4 Xor &HFF
       reception=48
   End Select

 ' ********** Mise à jour état des sorties *********
   delay 100
   Out 4,sortie1
   Out 5,sortie2
   Out 6,sortie3
   Out 7,sortie4

   If reception = 48 Then
     If sortie1=0 Then
       Putstr 1,"La sortie N° 1 est non active.",13
     Else
       Putstr 1,"La sortie N° 1 est active.",13
     End If
     Delay 100
     If sortie2=0 Then
       Putstr 1,"La sortie N° 2 est non active.",13
     Else
       Putstr 1,"La sortie N° 2 est active.",13
     End If
     Delay 100
     If sortie3=0 Then
       Putstr 1,"La sortie N° 3 est non active.",13
     Else
       Putstr 1,"La sortie N° 3 est active.",13
     End If
     Delay 100
     If sortie4=0 Then
       Putstr 1,"La sortie N° 4 est non active.",13,13
     Else
       Putstr 1,"La sortie N° 4 est active.",13,13
     End If
   End If
   Loop
```

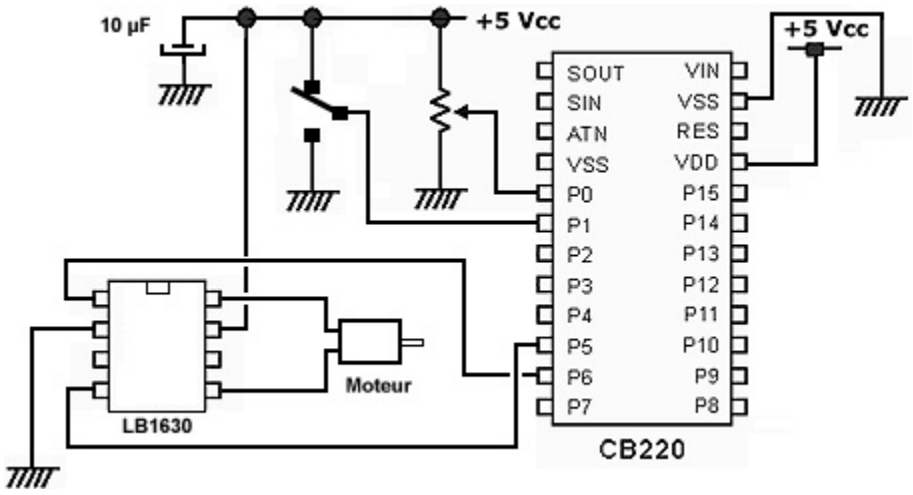
NOTE D'APPLICATION # 33.
Gestion d'un moteur « CC » à l'aide d'un LB1630

Cette note d'application va vous permettre de piloter un petit moteur « CC » au moyen d'un circuit intégré spécialisé « LB1630 ». Développé par Sanyo™, ce circuit est un contrôleur de moteur bi-directionnel faible saturation. Economique et très simple d'emploi, il vous permettra de réaliser des robots mobiles en un « rien de temps ». Ce circuit peut piloter des moteurs de 2,5 à 6 Vcc avec une consommation max. de l'ordre de 400 mA env. 2 entrées suffisent à sélectionner le sens de rotation et la vitesse du moteur. Le pont en « H » intégré dans le circuit dispose déjà de diodes de protection de telle sorte qu'un simple condensateur suffit à le rendre opérationnel.

Notions abordées :
- Gestion d'un signal PWM.

Matériel nécessaire :
- Une platine « CUBLOC Study Board » (facultative)
- 1 circuit intégré « LB1630 » + 1 petit moteur « CC ».

Cette note d'application mettra en œuvre le circuit « LB1630 » avec 2 exemples de programmes, lesquels utilisent le même schéma d'application ci-dessous.



Réalisez le montage en vérifiant la polarité du condensateur de 10 µF et en utilisant un moteur de faible consommation.

Saisissez ensuite le premier petit programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « LB1630 »).

```
'###################################
'#        Gestion d'un moteur « CC »         #
'#           à l'aide d'un « LB1630 »          #
'#     @Lextronic 2006 - 17/09/2006        #
'###################################

Const Device = CB220
Dim pot As Integer
Input 0                          ' Configure entrée de conversion "A/N" N°0 en entrée
Input 1                          ' Configure Port P1 en entrée
Low 5                            ' Configure port PWM 0 en sortie (avec niveau bas)
Low 6                            ' Configure port P6 en sortie (sens de rotation)
Do
   pot = Adin(0)                 ' Lecture de la postion du poInntiomètre
   If In(1) = 0 Then             ' Test position commutateur sens de rotation
      Out 6, 0                   ' Entree LB1630 (IN1) = 0
      Pwm 0,pot,1024             ' Génère signal PWM proportionnel à la position du potentiomètre
    Else
      Out 6, 1                   ' Entree LB1630 (IN1) = 1
      Pwm 0,1024-pot,1024        ' Génère signal PWM proportionnel à la position du potentiomètre
    End If
Loop
```

Interprétation du programme :
Ce premier programme vous permettra de sélectionner le sens de rotation du moteur à l'aide d'interrupteur et sa vitesse de rotation à l'aide d'un potentiomètre.

Suivant la position de l'interrupteur, on applique une polarité différente sur la broche 1 (IN1) du circuit « LB1630 » (afin de sélectionner le sens de rotation du moteur).

Conformément ànt la notice du « LB1630 » on sera alors obligé d'inverser ou non le rapport cyclique du signal PWM applique sur l'entrée 4 (IN2) du circuit pour obtenir la rotation du moteur.

Saisissez ensuite le second petit programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « LB1630A »).

```
###################################
'#        Gestion d'un moteur « CC »         #
'#          à l'aide d'un « LB1630 »          #
'#     @Lextronic 2006 - 17/09/2006        #
'###################################

Const Device = CB220
Dim pot As Integer
Input 0                          ' Configure entrée de conversion "A/N" N°0 en entrée
Input 1                          ' Configure Port P1 en entrée
Low 5                            ' Configure port PWM 0 en sortie (avec niveau bas)
Low 6                            ' Configure port P6 en sortie (sens de rotation)
 Do
    pot = Adin(0)                ' Lecture de la postion du potentiomètre
    If pot > 512 Then            ' Test position mediane du potentiometre
      Out 6, 0                   ' Entree LB1630 (IN1) = 0
      pot = (pot-512)*2
      Pwm 0,pot,1024             ' Génère signal PWM proportionnel à la position du potentiomètre
    Else
      Out 6, 1                   ' Entree LB1630 (IN1) = 1
      Pwm 0,pot*2,1024           ' Génère signal PWM proportionnel à la position du potentiomètre
    End If
 Loop
```

Interprétation du programme :
Ce second programme vous permettra de sélectionner le sens et la vitesse de rotation du moteur à l'aide d'un unique potentiomètre.

- Lorsque le potentiomètre est à mi-course, le moteur est à l'arrêt.
- Lorsque vous tournerez progressivement le potentiomètre vers la droite (depuis la position mi-course), le moteur tournera de plus en plus vite dans un sens.
- Lorsque vous tournerez progressivement le potentiomètre vers la gauche (depuis la position mi-course), le moteur tournera de plus en plus vite dans l'autre sens.

Pour ce programme, on teste la position mi-course de la valeur analogique recueillie au niveau du curseur du potentiomètre et en fonction de cette valeur, on applique une polarité différente sur la broche 1 (IN1) du circuit « LB1630 » (afin de sélectionner le sens de rotation du moteur).

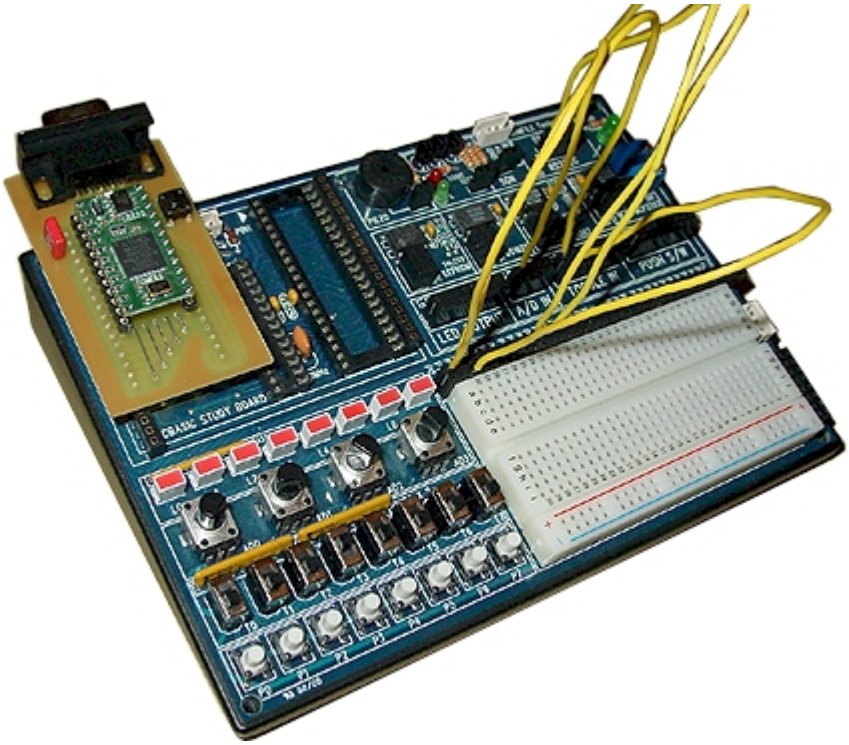
On ajuste alors la valeur du rapport cyclique du signal PWM pour obtenir une vitesse de rotation proportionnelle à la position du potentiomètre.

Note : Du fait qu'on exploite la « moitié » de la position du potentiomètre pour sélectionner la vitesse de rotation du moteur, la variation de vitesse sera moins progressive que dans l'exemple du programme N° 1.

NOTE D'APPLICATION # 34.
Circuit d'adaptation pour « PB Study Board »

Cette note d'application va permettre à tous les utilisateurs de modules « PICBASIC » ayant déjà fait l'acquisition d'une platine d'évaluation de type « PB Study Board » de pouvoir essayer le CB220 sans avoir à ré-investir dans une nouvelle carte de développement. Pour rappel, les « PICBASIC » correspondent à la génération précédente des produits développés par Comfile technology.

L'avantage à pouvoir réutiliser la platine « PB Study Board » est que cette dernière dispose d'une multitude de circuits intégrés permettant de disposer de fonctions complémentaires (horloge temps réel, convertisseurs « A/N » 12 bits, capteur de température, etc…



Nous vous proposons donc de réaliser un petit circuit d'interface qui viendra s'enficher sur 2 connecteurs de la platine de test. Ce circuit imprimé dispose d'un support pour le « CB220 », d'un BP Reset et d'une prise Sub-D 9 broches pour la programmation du « CB220 ». Une fois inséré sur la platine « PB Study Board », toutes les broches du « CB220 » sont accessibles sur la barrette en haut de la plaque de connexion (P0 à P15).

Le dessin du circuit imprimé a été réalisé à l'aide du logiciel « Sprint Layout ». Vous pourrez trouver sur le CD-ROM Lextronic ou sur notre site Internet : www.lextronic.fr le fichier « source » pouvant être visualisé avec la version de démo de « Sprint Layout » ou modifié sur la version commerciale.
Le logiciel est disponible sur notre site Internet ou sur le site du fabricant : http://www.abacom-online.de/UK/default.html
Le circuit a été réalisé en double face, toutefois il vous sera également possible de remplacer les pistes de la seconde face par des straps (vous permettant ainsi de réaliser la carte en simple face).

Une fois le montage réalisé, insérez le circuit sur le support de la platine « PB Study Board » en s'assurant impérativement que la platine est hors tension !

A noter qu'en fonction du nombre de demande, nous pourrons être amenés à commercialiser ce circuit imprimé dans les semaines à venir. Consultez-nous.

NOTE D'APPLICATION # 35.
Gestion d'un modem radio « Xbee™ »

Cette note d'application va vous permettre de piloter un Modem radio « Xbee™ » du fabricant Maxstream™.
Ce dernier est un modem basé sur une technologie ZigBee™ Il intègre une "électronique intelligente" qui vous permettra de réaliser aussi bien des communications "basiques" point-à-point que de véritables réseaux de transmission de données multi-points. Il assurera la transmission à distance de signaux numériques séries de façon totalement transparente. D'un point de vue utilisation celui ci s'apparente en fait à un simple câble série « virtuel » très rapide et simple à mettre en œuvre.

ZigBee™ is a registered trademark of the ZigBee Alliance

Notions abordées :
- Gestion communications séries.

Matériel nécessaire :
- Une platine « CUBLOC Study Board » (facultative)
- 2 modules « Xbee™ » (ou un module « Xbee™» et un module « XBP24PKS »

Cette note d'application vous montera comment interfacer ce modem avec un CB220 afin de pouvoir réaliser une télécommande radio 4 canaux bidirectionnelle (via un compatible PC) avec fonction d'accusé de réception des ordres. Le module « Xbee™ » dispose d'une interface série (niveau logique 0 / + 3,3 V qui pourra très facilement s'interfacer avec le CUBLOC). Du côté du PC, il vous faudra ajouter une circuit de mise à niveau **MAX-3232** (compatible 3,3 V) afin d'adapter les signaux +/-12 V du port RS-232 avec les niveaux du « Xbee™ » (vous trouverez un schéma d'application « type » sur le site du fabricant : www.maxstream.net).

Il est également possible d'avoir recours à un module en boîtier « XBP24PKS », lequel intègre sur une même platine le modem radio, une prise SUB-D 9 broches, son antenne et le circuit d'interfaçage Max-3232 (de telle sorte qu'il vous suffira simplement de le relier au port RS-232 du PC pour le rendre opérationnel).

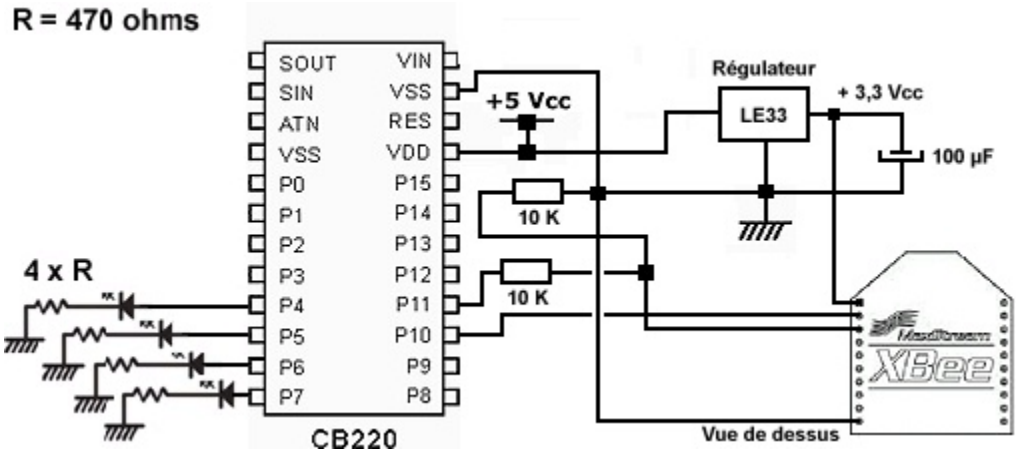
Préparation matérielle :

Du côté du PC, celle-ci consiste à relier le boîtier « XBP24PKS » à un port COM de libre du PC et à exécuter le programme de communication « HyperTerminal™ ». Ce programme se trouve via les menus :

Démarrer -> **Programmes** -> **Accesoires** -> **Communications** -> **HyperTerminal.**

Une fois le programme exécuté, donnez un nom à la communication (par exemple CUBLOC), puis sélectionnez le N° de port sur lequel est connecté le module « XBP24PKS » dans l'onglet du bas de la fenêtre. Validez par OK. Sélectionnez enfin une vitesse de connexion de 9600 bds / 8 bits / Parité (aucun) / 1 bit de stop / Contrôle de flux (aucun) et validez à nouveau. Cliquez ensuite sur l'onglet « Paramètres », puis cliquez sur « Configuratio ASCII… » et validez la sélection « Ajouter les changements de ligne à la fin des lignes entrantes ». Alimentez alors le module « XBP24PKS » à l'aide d'un bloc d'alimentation secteur 220 VAC/12Vcc.

Du côté du module CUBLOC, il vous faudra réaliser le montage ci-dessous. Un régulateur 3,3 V sera nécessaire pour alimenter le module « Xbee™ » à partir du + 5 V général (le condensateur de filtrage doit être impérativement utilisé). Un pont diviseur sera également nécessaire pour limiter la tension de la sortie TX à 2,5 V max. afin de ne pas endommager le modem radio.



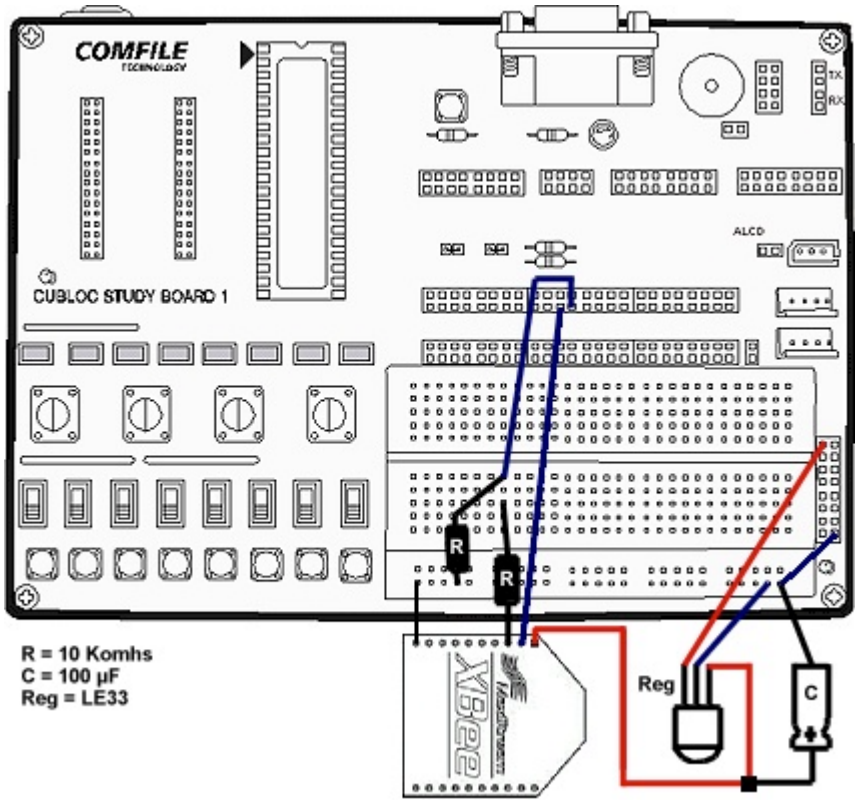
Principe de fonctionnement de l'application :

Le programme HyperTerminal™ présent sur le PC vous permettra d'envoyer des ordres séries depuis le clavier de votre PC, lesquels seront transmis au module CUBLOC via les 2 modems radio. Ainsi en sollicitant les touches 1 à 4 du PC, vous changerez alternativement l'état des 4 sorties du CUBLOC (une sollicitation de la touche 1 du PC allume la Led de la première sortie du CUBLOC – une seconde sollicitation de la touche 1 du PC éteint la Led de la première sortie et ainsi de suite pour les touches 2 à 4 et les sorties Leds 2 à 4 du module CUBLOC).

Le programme HyperTerminal™ peut également recevoir des données en provenance du port sortie en les affichant à l'écran. La note d'application utilise cette possibilité pour renvoyer après chaque commande l'état des 4 sorties du CUBLOC. Ainsi après avoir sollicité une des touches 1 à 4 au niveau du PC, le CUBLOC change l'état de la sortie adéquat et renvoie sous forme de textes l'état des 4 sorties qui s'affichera sur la fenêtre du PC (vous serez ainsi sûr que l'ordre radio a correctement été reçu et que la sortie a bien l'état que vous vouliez lui faire prendre).

En sollicitant la touche 0 du PC, le CUBLOC ne modifie aucun état de ses sorties, mais renvoie tout de même l'état de ces dernières. La touche 0 fait ainsi office de touche d'interrogation en quelque sorte (dans le cas de figure où vous avez un doute sur l'état des sorties du module CUBLOC).

Réalisation du montage à l'aide de la platine « Cubloc Study Board » :



Afin de faciliter la mise en œuvre du modem radio côté CUBLOC, il vous sera possible d'utiliser une platine «CUBLOC Study Board» comme le montre le schéma ci-dessus. Le module « Xbee™ » dispose de broches de connexion au pas de 2 mm. Vous pourrez utiliser un connecteur de référence « CON-EZL » qui vous permettra de relier ce dernier à la platine d'essai.

Saisissez ensuite le petit programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « Xbee »).

Interprétation du programme :
Le programme commence par l'initialisation des variables représentant l'état des 4 sorties du module CUBLOC (et de leur initialisation « physique » au niveau logique 0 V). Le programme continu ensuite par l'analyse du contenu du buffer série de réception du module CUBLOC. Si le CUBLOC a reçu un code ASCII correspondant à une des touches 1 à 4 du PC, il s'en suit un changement d'état intermittent (via une fonction XOR) des sorties en fonction du caractère reçu dans le buffer. La valeur 49 correspond à la valeur ASCII du chiffre de la touche 1 sollicitée sur le PC. La valeur 50 correspond au chiffre 2, etc…Une fois la variable de la sortie remise à jour, le programme force la variable de réception avec la valeur 48 (correspondant à la valeur ASCII du chiffre de la touche 0 du clavier – ceci afin de « forcer » le reste du programme à renvoyer un accusé de réception). Le programme remet ensuite à jour l'état « physique » des sorties puis envoi un accusé de réception de l'état de chaque sortie via des phrases dont chaque lettre est transmise au module radio « Xbee™ » afin que ces phrases s'affichent dans la fenêtre de HyperTerminal™ du PC.

NOTE IMPORTANTE :
Il est important de signaler que le programme de cette note d'application ne fait office que de programme d'évaluation et de test et qu'il ne devra en aucun cas être utilisé tel quel au sein d'une application. En effet, les sorties du CUBLOC sont directement activées via la réception d'un seul octet. Dès lors n'importe quel autre module « X24-009 » émettant à proximité de votre application pourra être susceptible de déclencher vos sorties si ce dernier envoi un code ASCII correspondant aux touches 1 à 4. Dans le cadre d'une application réelle, il vous faudra impérativement envoyer au préalable un code sur plusieurs octets correspondant à une sorte de « mot » de passe ou de « clef de sécurité » (ce code pourra être répété avant et après les ordres de commande). Dès lors le CUBLOC devra avant toute activation de ses sorties s'assurer de la validité de ce code afin que vous soyez sûr qu'il s'agisse bien de votre application qui essai de vous envoyer un ordre radio.

```
'###################################
'#        Gestion d'un modem radio          #
'#          Maxstream™ "Xbee™"              #
'#     @Lextronic 2006 - 17/09/2006         #
'###################################

Const Device = CB220

Dim sortie1 As Byte
Dim sortie2 As Byte
Dim sortie3 As Byte
Dim sortie4 As Byte
Dim reception As Byte

Opencom 1,9600,3,10,35

'****************** RAZ sorties *****************
Low 4
Low 5
Low 6
Low 7
Sortie1 = 0
Sortie2 = 0
Sortie3 = 0
Sortie4 = 0

'***************** Boucle principale *************
Do
   reception=Get(1,1)
   Select Case reception
     Case 49
       sortie1=sortie1 Xor &HFF
       reception=48
     Case 50
       sortie2=sortie2 Xor &HFF
       reception=48
      Case 51
       sortie3=sortie3 Xor &HFF
       reception=48
     Case 52
       sortie4=sortie4 Xor &HFF
       reception=48
   End Select

 ' ********** Mise à jour état des sorties *********
    delay 100
    Out 4,sortie1
    Out 5,sortie2
    Out 6,sortie3
    Out 7,sortie4

    If reception = 48 Then
      If sortie1=0 Then
        Putstr 1,"La sortie N° 1 est non active.",13
      Else
        Putstr 1,"La sortie N° 1 est active.",13
      End If
      Delay 100
      If sortie2=0 Then
        Putstr 1,"La sortie N° 2 est non active.",13
      Else
        Putstr 1,"La sortie N° 2 est active.",13
      End If
```

```
    Delay 100
    If sortie3=0 Then
      Putstr 1,"La sortie N° 3 est non active.",13
    Else
      Putstr 1,"La sortie N° 3 est active.",13
    End If
    Delay 100
    If sortie4=0 Then
      Putstr 1,"La sortie N° 4 est non active.",13,13
    Else
      Putstr 1,"La sortie N° 4 est active.",13,13
    End If
  End If
Loop
```

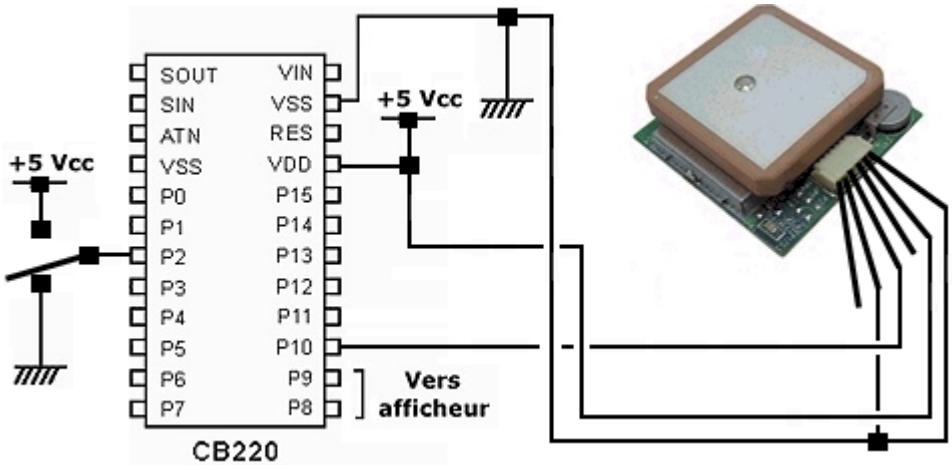
NOTE D'APPLICATION # 36.
Décodage de trames « GPS »

Cette note d'application va vous permettre de décoder des trames « GPS » au format NMEA-0183 afin de pouvoir afficher les informations telles que la Longitude, la Latitude, l'heure UTC, le nombre de satellites « en poursuite » ainsi que la « force » des signaux des 8 premiers satellites en acquisition et enfin les sattelites dont les signaux sont valides pour l'acquisition. Pour ce faire on utilisera un module GPS OEM subminiature « EM-406 » avec antenne amplifiée intégrée. Ce module délivre un signal série au format NMEA-0183 qui se traduit sous la forme de trames de caractères ASCII .

Notions abordées :
- Gestion communication série

Matériel nécessaire :
- Une platine « CUBLOC Study Board » (facultative)
- Module « EM-406 »

Préparation matérielle :
Celle-ci repose sur le schéma théorique ci-dessous.



Le module GPS dispose d'un connecteur associé à 6 fils de raccordement avec un second connecteur en bout. Il suffira de couper ce second connecteur et de souder les fils pour y raccorder l'alimentation et la sortie série du signal du GPS vers le module CUBLOC (déconnectez le module GPS pendant l'opération de soudage).

Saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « gps »).

Principe de fonctionnement de l'application :
Le programme consiste à extraire et mémoriser certaines trames du signal série disponible en sortie du module GPS et à « décoder » les informations qui y sont présentes pour récupérer les données qui nous intérressent.

Pour ce faire, nous nous intéresserons aux trames :

GGA : comprenant entre autre l'heure UTC, la longitude, la latitude et le nombre de satellites en poursuite.

GSA : comprenant entre autre les N° ID des satellites utilisés pour le FIX (les satellites dont les données sont valides).

GSV : comprenant entre autre les N° ID de tous les satellites et de la force de leur signaux.

Nota : Du fait que les données soient affichées sur un écran LCD 4 x 20 caractères, nous nous contenterons d'afficher uniquement la force des signaux des 8 premiers satellites à l'aide de bargraphs (pour info le module GPS est à même de restituer les signaux de 12 à 20 satellites).

Le programme commence initialiser un port de communication à 4800 bds (débit utilisé par le GPS). Puis le programme efface l'écran et affiche la valeur de la force des signaux des 8 premiers satellites (les valeurs sont préalablement initialisées à 000).

Les valeurs de la force des signaux des 8 premiers satellites seront stockées dans un tableau à 2 dimensions : SIGNAL (4,2) - la première colonne correspondra à la force des signaux des satellites de 1 à 4 et la deuxième colonne à celle des satellites de 5 à 8.

Les numéros d'identifications (ID) des 8 premiers satellites seront également stockés dans un tableau à 2 dimensions : ID (4,2) - la première colonne correspondra au N° d'ID des satellites de 1 à 4 et la deuxième colonne à ceux des satellites de 5 à 8.

Un autre tableau « FIX » sera utilisé pour stocker les N° ID des satellites utilisés pour le FIX (c'est à dire des satellites dont les données sont exploitables pour le « GPS »).

Le programme continu ensuite en réalisant une boucle sans fin (do – loop) dans laquelle le module CUBLOC attend de récupérer successivement les caractères « S » et « A » de son buffer série (afin de détecter les lettres « SA » de a trame « GSA ». Si c'est le cas, le CUBLOC « sort » de cette boucle sans fin et attend de mémoriser les 30 caractères suivant en provenance du module GPS dans la variable « donnee » afin de récupérer ainsi le contenu de la trame « GSA ».

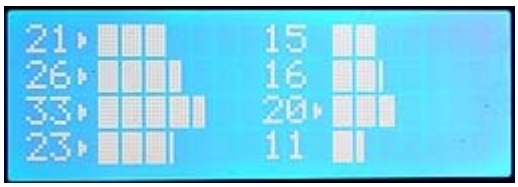
Dès lors, le programme va récupérer les N° ID des 8 premiers satellites utilisés pour le FIX (c'est à dire des satellites dont les données sont exploitables par le « GPS ») en « détectant » les virgules mémorisées dans la chaîne « donnee » et en vérifiant s'il y a des n° d'ID après ces virgules (si ce n'est pas le cas, le N° d'ID est initialisé avec la valeur « 0 »). Durant cette procédure, le programme mémorise les n° d'ID dans la variable tableau « fix ».

Le programme va ensuite décoder les trames « GPGSV » (il y a 3 trames « GPGSV » de suite qui se succèdent lorsque le module « GPS » restitue les signaux de 12 satellites). Chaque trame délivre des informations sur 4 satellites. Nous décoderons ainsi 2 trames « GPGSV » pour disposer des informations des 8 premiers satellites. On aura ainsi recours à un sous-programme qui servira à simplifier le programme.

Ce sous-programme réalise une boucle sans fin (do – loop) dans laquelle le module CUBLOC attend de récupérer le caractère « V » (correspondant à la trame « GPGSV »). Si c'est le cas, le CUBLOC « sort » de cette boucle sans fin et attend de mémoriser les 5 autres caractères suivant en provenance du module GPS avant de vérifier si le dernier caractère mémorisé correspond bien au N° de la trame « GPGSV » 1 ou 2 (ceci afin de savoir dans quelle colonne des tableaux mémoire on devra mémoriser les données).

Une fois cette opération effectuée, le CUBLOC attend de mémoriser les 75 caractères suivant en provenance du module GPS dans la variable « donnee » afin de récupérer ainsi le contenu de la trame « GPGSV ». Un sous-programme appelé plusieurs fois va alors récupérer les N° ID et les valeurs des signaux de réception de chaque satellite en « détectant » les virgules mémorisées dans la chaîne « donnee » et en récupérant les données qui sont derrière. Durant cette procédure, le programme mémorise ces information dans les variables tableaux « signal » et « id ».

Une fois toutes les données collectées, le programme passe à la phase d'affichage.



Celle-ci se « passe » dans une sous-routine qui va successivement afficher la valeur de la force de réception du signal de chacun des 8 satellites associée à une représentation sur un bargraph horizontal. La réalisation de ce bargraph nécessite la reconfiguration de caractère de l'afficheur afin de pouvoir disposer d'une « bonne » résolution d'affichage. L'explication du fonctionnement du bargraph est disponible sur la note d'application N° # 01 – Consultez notre site Internet ou notre CD-ROM).

La dernière opération réalisée par la sous-routine consiste à afficher un petit triangle devant le bargraph du (des) sattelite(s) utilisé(s) pour le fix (c'est à dire pour lequel(s) les données sont exploitables). Pour ce faire le programme va récupérer les valeurs mémorisées dans le tableau (fix) et va vérifier si ces valeurs correspondent à celle mémorisées dans le tableau (id). Si c'est le cas, ces que les données sont « valides » et on affiche un petit triangle.

Note : Pour cette application, nous avons choisi par mesure de simplicité de mémoriser et décoder les trames 1 et 2 « GPGSV » les unes après les autres. Ceci signifie que le temps d'acquisition total et de rafraîchissement des bargraphs est relativement « lent » (environ toutes les 10 secondes). De même un décalage entre le « marquage » (par le triangle) des satellites utilisés pour le FIX et la valeur de la force de leur signaux pourra se faire ressentir dans certains cas (car une fois encore les trames sont acquises et décodées les unes après les autres)

L'application idéale étant celle qui consisterait à mémoriser toute la trame NMEA-0183 et à décoder ensuite successivement les données. Mais il nous a parut plus « simple à expliquer » de procéder au décodage individuel de chaque trame.

L'ensemble de traitement décrit ci-avant est réalisé dans une boucle sans fin qui est stoppée suivant le niveau logique appliqué sur le port P2 du CUBLOC. Ceci permet d'utiliser un interrupteur qui lorsqu'il est mis à la masse vous permettra de « passer » à un second écran d'information (note : Lors de la manipulation de l'interrupteur, le second écran apparaîtra avec un délai pouvant aller jusqu'à 10 secondes – délai correspondant au temps pris par le CB 220 pour faire l'acquisition en cours des trames « GPGSV » (voir remarque ci avant). Encore une fois, il est important de signaler que le traitement opéré lors du décodage des trames du « GPS » par le CUBLOC dans cette note d'application est très simple et ne prend pas en charge le décodage du checksum des trames, ni par exemple la vérification de l'arrivée du signal série du GPS (si ce dernier s'interromps pour une raison ou une autre, le CB220 attendra en permanence les trames NMEA-0183 pour les décoder).

Le deuxième écran pour permettra d'afficher l'heure UTC, la longitude, la latitude et le nombre de satellites en poursuite grâce au décodage de la trame « GPGGA ».

Ce décodage est réalisé dans une autre boucle sans fin « do – loop » dont l'arrêt (pour le retour au premier écran) est également conditionné par la position de l'interrupteur du port P2.

Pour ce faire, le programme attend de récupérer successivement les caractères « GGA » (correspondant à la trame « GPGGA »). Si c'est le cas, le CUBLOC attend de mémoriser les 75 caractères suivant en provenance du module GPS dans la variable « donnee » afin de récupérer ainsi le contenu de toute la trame « GPGGA ». Le programme va ensuite s'assurer que la valeur de la latitude n'est pas nulle (ou absente – ce qui se traduit par la présence d'une « virgule » à la place. Si c'est le cas, le programme n'affiche alors acune donnée exploitable (celles-ci sont remplacées par des « tirets »).

A l'inverse, le programme affichera alors l'heure UTC, la longitude, la latitude et le nombre de satellites en poursuite.

Le programme doit normalement pouvoirfonctinner avec la plupart des modules « GPS » doté d'un format de sortie NMEA-0183. Mais celui-ci n'a été testé que sur le module « EM-406 ».

```
'###################################
'#       Gestion d'un récepteur GPS       #
'#                « EM-406 »                #
'#    @Lextronic 2006 - 22/09/2006      #
'###################################

Const Device = CB220

Dim reception As Byte, trame As Byte
Dim a As Byte, x As Byte, y As Byte
Dim donnee As String * 80
Dim signal (4,2) As String *3                    ' Force du signal
Dim id (4,2) As String * 3                       ' No ID satellite en acquisition
Dim fix (9) As String * 3                        ' No ID Satellites utilise pour le FIX

Dim i As Byte
Dim J As Byte
Dim valmax As Single
Dim barre As String*20
Dim vide As String*20
Dim segment As String*1

Input 2                                          ' Configure port 2 en entree

Opencom 1,4800,3,80,5

Set Display 2,0,1,50

Print &H1B,&H44,8,0,0,0,0,0,0,0                  ' Redéfénition des caractères du LCD
Print &H1B,&H44,9,16,16,16,16,16,16,16,16
Print &H1B,&H44,10,24,24,24,24,24,24,24,24
Print &H1B,&H44,11,28,28,28,28,28,28,28,28
Print &H1B,&H44,12,30,30,30,30,30,30,30,30
Print &H1B,&H44,13,00,00,8,12,12,8,00,00


'********************** Initialisation donnees gps en memoire ***************

Cls
Delay 300
Csroff
For x = 0 To 1                                   ' Initialise affichage
  For y = 0 To 3
    signal (y,x) = "00"
    id (y,x) = "00"
  Next
Next
For x = 0 To 8
  fix (x) = "99"
Next
```

```
Do

'************** Acquisition trame GPGSA et memorisation ID FIX ****************

  Do
   Gosub aff_val                                  ' Affichage des donnees / bargraph sur le LCD
          Do
            Do
              Do
                reception=Get(1,1)
       If reception = &h53 Then Exit Do           ' Attend caractère "S"
     Loop
     Do While Blen(1,0)<2
     Loop
     reception=Get(1,1)
     If reception = &h41 Then Exit Do             ' Attend caractère "A"
   Loop
   Do While Blen(1,0)<30                          ' Attend que le buffer serie
   Loop                                           ' dispose d'au moins 30 octets
   donnee = Getstr(1,30)                          ' puis les récuprere
   Exit Do
  Loop
  For x = 0 To 7                                  ' RAZ valeur No id des fix
   fix(x) = "00"
  Next
  x = 0                                           ' RAZ tableau FIX
  reception = 6                                   ' On se positionne sur 8eme caractere trame
  a = 0                                           ' initialise Nb de virgules a detecter
          Do
            If Mid(donnee,reception,1) = "," Then ' Regarde si on a detecté une virgule ?
              a = a + 1                           ' OUI -> memorise une virgule de plus
     reception = reception + 1                    ' Passe au caractere suivant dans la trame
   Else
     fix(x) = Mid (donnee,reception,2)            ' Memorise No ID d FIX
     reception = reception + 2
     x = x + 1                                    ' Incremente valeur du tableau
   Endif
   If a = 8 Then Exit Do                          ' A t'on detecté 8 virgules ?

  Loop


'************************* Gestion decodage trame GPGSV ************************

  y = 0                                           ' Selectionne 1ere ligne dans tableau memoire
  trame = &h31                                    ' Selectionne 1ere trame GPGSV
  Gosub dec_gpgsv                                 ' Va decoder 1ere trame GPSV


'********************************** Test si  passage autre ecran  *****************

  If In(2) = 0 Then                               ' Test position interrupteur
    Exit Do
  Endif

  y = 1                                           ' Selectionne 2eme ligne dans tableau memoire
  trame = &h32                                    ' Selectionne 2ere trame GPGSV
  Gosub dec_gpgsv                                 ' Va decoder 2ere trame GPSV
   Loop
```

```
'******************************** Decodage trame GPGGA *************************

Cls                                          ' Efface l'ecran
Delay 300
Csroff
Locate 0,0
Print "Heure UTC"
Locate 0,1
Print "Lat"
Locate 0,2
Print "Lon"
Locate 0,3
Print "id       Nb Sat.""

Do
   Do
      Do
         reception=Get(1,1)
         If reception = &h47 Then Exit Do
      Loop
      Do While Blen(1,0)<5
      Loop
     donnee = Getstr(1,5)
     If Mid(donnee,3,1)="G" And Mid(donnee,4,1)="A" Then Exit Do
   Loop
 Do While Blen(1,0)<75                                  ' Attend que le buffer serie
 Loop                                                   ' dispose d'au moins 75 octets
 donnee = Getstr(1,75)                                  ' puis les récuprere
 If In(2) = 1 Then Exit Do                              ' Test position interrupteur

 '------ Regarde si il y a au moins 3 satellites exploitables ----------------

  a = 0                                                 ' Initialise Nb de virgules a detecter
  reception = 1                                         ' initialise position dans la chaine
  Do
     If Mid(donnee,reception,1) = "," Then a = a + 1    ' memorise une virgule de plus
        reception = reception + 1                       ' Passe au caractere suivant dans la trame
        If a = 1 Then Exit Do                           ' A t'on detecté la premiere virgule ?
  Loop
  If Mid(donnee,reception,1) <> "," And Val(Mid(donnee,reception,2)) <> 0 Then       ' Test si donnee trame OK ?
    Locate 11,0                                                 ' Affiche heure UTC
    Print Left(donnee,2),":",Mid(donnee,3,2),":",Mid(donnee,5,2)
    Locate 4,1                                                  ' Affiche Latitude
    Print Mid(donnee,12,2)," deg ",Mid(donnee,14,7),"'",Mid(donnee,22,1)
    Locate 4,2                                                  ' Affiche Longitude
    Print Mid(donnee,25,2)," deg ",Mid(donnee,27,7),"'",Mid(donnee,35,1)
    Locate 4,3
    Print "OK"                                                 ' Affiche Fix qualification OK
    Locate 18,3                                                 ' Affiche Nb sattelites en poursuite
    Print Mid(donnee,39,2)
  Else
    Locate 11,0
    Print "--:--:--"
            Locate 4,1
            Print "-- deg ------- -"
            Locate 4,2
            Print "-- deg ------- -"
    Locate 4,3
    Print "--"
    Locate 18,3
    Print "00"
  Endif
  Loop
 Loop
```

```
'************************* Decodage trame GPGSV ************************

        dec_gpgsv:
        x = 0                                                   ' Initialise colonne tableau memoire
        Do
          Do
            reception=Get(1,1)
   If reception = &h56 Then Exit Do     ' Attend caractère "V"
  Loop
  Do While Blen(1,0)<5
  Loop
  reception=Get(1,1)
  reception=Get(1,1)
  reception=Get(1,1)
  reception=Get(1,1)
          If reception = trame Then                             ' Regarde si No trame GPGSV a decoder ?
            Do While Blen(1,0)<75                               ' Attend que le buffer serie
    Loop                                                        ' dispose d'au moins 75 octets
    donnee = Getstr(1,75)                                       ' puis les récuprere
  Exit Do
  Endif
 Loop
 reception = 2                                                  ' initialise position dans la trame recue
 Gosub dec_sign                                                 ' Va recuperer 1ere valeur du signal
 x = x + 1                                                      ' Selectionne signal suivant
 Gosub dec_sign                                                 ' Va recuperer 2eme valeur du signal
 x = x + 1                                                      ' Selectionne signal suivant
 Gosub dec_sign                                                 ' Va recuperer 3eme valeur du signal
 x = x + 1                                                      ' Selectionne signal suivant
 Gosub dec_sign                                                 ' Va recuperer 4eme valeur du signal
 Return


'********* Sous-routine recuperation force signal + No ID Satellite ********

        dec_sign:
        a = 0
        ' initialise Nb de virgule a detecter
        Do
          If Mid(donnee,reception,1) = "," Then a = a + 1       ' memorise une virgule de plus
  reception = reception + 1                                     ' Passe au caractere suivant dans la trame
  If a = 1 Then Exit Do                                         ' A t'on detecté la premiere virgule ?

 Loop
 id(x,y) = Mid (donnee,reception,2)                             ‘ Memorise No ID du satellite
        Do
          If Mid(donnee,reception,1) = "," Then a = a + 1       ' Memorise une virgule de plus
  reception = reception + 1                                     ' Passe au caractere suivant dans la trame
  If a = 4 Then Exit Do                                         ' A t'on detecté les 3 autres virgules ?

 Loop

signal (x,y) = Mid (donnee,reception,2)
If Left(signal(x,y),1) = "," Or Left(signal(x,y),1) = "*" Then signal(x,y) = "00"
Return


'********** Sous-routine d'affichage des donnees / bargraph ******************

 aff_val:
 For x = 0 To 1                                                 ' Affichage valeur force du signal
          For y = 0 To 3
   Locate (x*10),y
```

```
    Print signal(y,x)
    a = Val(signal(y,x))

'--- Generation du bargraph

    valmax=(a/1.8)/5
    i=valmax
    barre=String(&HFF,i)
    J=6-i
    vide=Spc(J)
    i=((valmax-i)/0.2)+8
    segment=Chr(i)
    barre=barre+segment+vide
    Locate 3+(x*10),y
    Print barre
    Next
  Next

'--- Pointage sur le LCD des satellite utilises pour le FIX

  For x = 0 To 1                                    ' Verification de toutes les adresses ID
            For y = 0 To 3
               For a = 0 To 7
                  Locate (x*10)+2,y
                  If id (y,x) = fix(a) Then         ' Regarde si ID fait parti des FIX ?
             Print 13                               ' Affichage caractere signal FIX OK
             Exit For
          Else
             Print " "
          Endif
      Next
    Next
  Next
  return
```

NOTE D'APPLICATION # 37.
Pilotage d'une base robotique « DS-X4 »

Cette note d'application va vous permettre de Piloter une base robotique « DS-X4 » à partir D'un module CB220. Livrée sous forme de kit à assembler, cette base tout terrain est équipée de 4 roues motrices avec pneus gommes, capables d'emprunter des chemins accidentés et très inclinés. Conçus sur la base d'un châssis aluminium solide, elle vous permetra la réalisation de robots ludiques très performants.

La motricité de ces bases est assurée par 4 servomoteurs indépendants type modélisme (spécialement modifiés pour qu'ils puissent tourner sur 360°) associés à des pneus de diamètre 75 mm x 28 mm de largeur. Ces servomoteurs sont pilotés par une petite platine électronique (livrée toute montée) qui pourra recevoir des ordres de commande séries (9600 bds) en provenance du CUBLOC). La petite platine électronique livrée avec la base robotique est également équipée de capteurs infrarouges permettant de détecteur la présence d'obstacles. Dès lors par le biais d'ordres très simples (caractères ASCII), votre électronique de commande pourra ordonner à la base:

- D'avancer tout droit (avec 9 vitesses au choix).
- De reculer tout droit (avec 9 vitesses au choix).
- De tourner à droite (avec 9 vitesses au choix).
- De tourner à gauche (avec 9 vitesses au choix).
- De stopper.
- D'activer les capteurs infrarouges pour détecter les obstacles.
- De désactiver le fonctionnement des capteurs infrarouges.

Lorsque les capteurs infrarouges sont activés et que la base rencontre un obstacle (pouvant réfléchir les rayons infrarouges), la base stoppe, recule (pendant une durée paramétrable de 1 à 9 sec.) puis tourne dans le sens opposé à l'obstacle (pendant une durée paramétrable de 1 à 9 sec.).

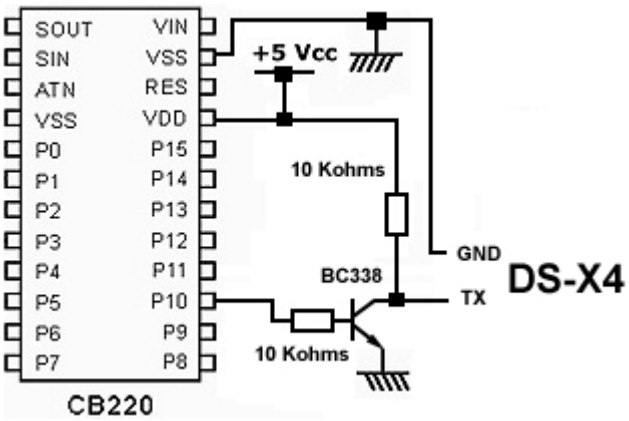
Un mode "test" permet également d'automatiser entièrement le déplacement de la base qui avancera en permanence tout en évitant les obstacles. Dans tous les cas la platine intégrée à la base prendra automatiquement en charge la génération des impulsions "PWM" nécessaires au pilotage des 4 servomoteurs.

Notions abordées :
- Gestion communication série

Matériel nécessaire :
- Un module « CUBLOC CB220 »
- Une base robotique « DS-X4 »
- 2 résistances de 10 Kohms
- 1 transistor BC338

Préparation matérielle :

Celle-ci repose sur le schéma théorique ci-dessous pour lequel on utilisera un transistor pour inverser la polarité du signa lde sortie du CUBLOC afin de le rendre compatible avec la base robotique « DS-X4 ».



Saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « ds-x4 »).

Principe de fonctionnement de l'application :

Le programme est très simple et consiste à envoyer des ordres séries pour faire exécuter différents mouvements à la base robotique.

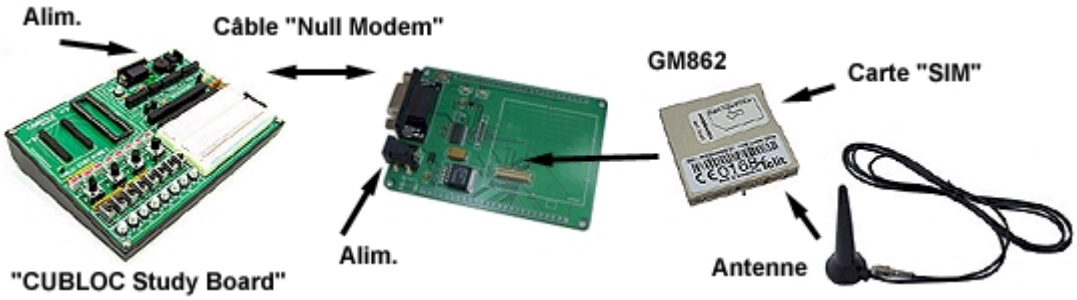
```
'###################################
'#        Gestion d'une base robotique      #
'#                « DS-X4 »                  #
'#    @Lextronic 2006 - 01/10/2006       #
'###################################

Const Device = CB220
Opencom 1,9600,3,30,20
Delay 1000
Putstr 1,"C0"                          ' Stoppe les capteurs
Do
   Putstr 1,"F5"                       ' Marche avant
   Delay 3000
   Putstr 1,"R1"                       ' Tourne à droite lentement
   Delay 5000
   Putstr 1,"F5"                       ' Marche avant
   Delay 3000
   Putstr 1,"L1"                       ' Tourne à gauche lentement
   Delay 5000
   Putstr 1,"F5"                       ' Marche avant
   Delay 3000
   Putstr 1,"S0"                       ' Stop
   Delay 2000
   Putstr 1,"B5"                       ' Marche avant
   Delay 3000
   Putstr 1,"S0"                       ' Stop
   Delay 2000
Loop
```

NOTE D'APPLICATION # 38.
Transmetteur d'alarme « GSM » expérimental

Cette note d'application va vous permettre de réaliser un transmetteur d'alarme « GSM » sans fil expérimental. Pour ce faire nous utiliserons un module GSM/GPRS « GM862 » "Quad-band" capable de supporter les modes de communication "voix", "fax", "Data", "SMS". Ce dernier se caractérise par ses faibles dimensions associées à un prix très compétitif et à une grande simplicité de mise en oeuvre. L'application permettra à un module CUBLOC « CB220 » de surveiller l'état de 2 entrées « tout-ou-rien » et d'une entrée analogique et (sur changement d'état des entrées numériques, d'envoyer un « SMS » d'alerte vers un portable afin de lui transmettre l'état des entrées).

Afin de simplifier la réalisation, on aura recours à une platine de test spécialement conçue pour pouvoir recevoir le module GSM « GM862 ». Cette platine comprend un étage de régulation ainsi qu'un étage de mise en forme du signal de sortie du module « GSM » (afin que les niveaux logiques de celui-ci soit compatibles avec une liaison RS232. Il nous suffira alors de connecter une alimentation à cette platine, une antenne au module GSM et d'utiliser un « NULL-MODEM » pour relier cette carte à la platine de test « CUBLOC Study Board » via leur port série.

La dernière opération indispensable consistera à se munir bien évidemment d'une carte « SIM » opérationnelle que l'on aura inséré (hors alimentation) dans le module « GM862 » grâce au connecteur prévu à cet effet. Le code « PIN » de la carte SIM devra être effacer afin de simplifier la procédure d'utilisation du module GSM.



On utilisera également de préférence 2 alimentations séparées (une pour la platine du GSM et une pour la platine « PB Study Board ».

Egalement libre à vous de réaliser directement votre propre platine support pour le module « GM862 ».

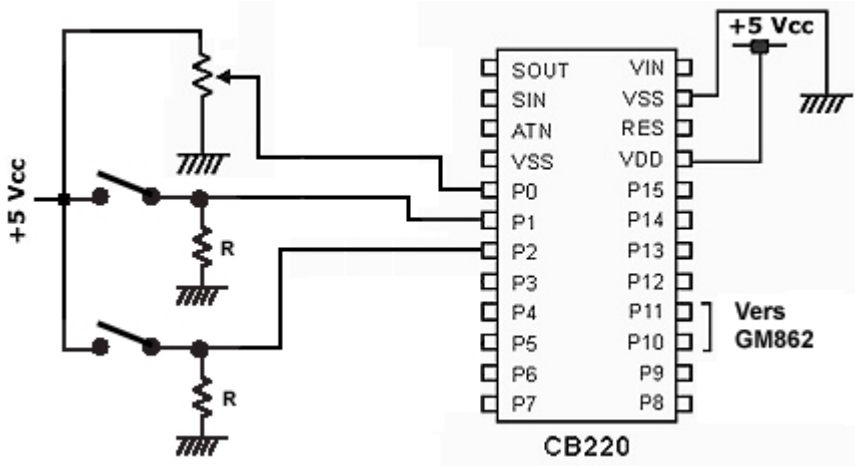
Notions abordées :
- Gestion communication série

Matériel nécessaire :
- Un module « CUBLOC CB220 »
- Une platine « PB Study Board »
- Un module « GM862 »
- Une antenne GSM
- Une platine d'interface pour « GM862 »
- 2 blocs d'alimentation
- 1 câble « Null Modem »
- 1 carte SIM opérationnelle (**avec code PIN effacé**).

Préparation matérielle :

Celle-ci repose en fait sur un simple raccordement entre le port série du « CB220 » et celui du module « GM862 » et au câblage d'un potentiomètre et de 2 interrupteurs sur les 3 premières entrées du « CB220 ».

Si vous ne passez pas par les platines de tests évoquées ci-avant et que vous reliez directement le module « GSM » au CUBLOC, vérifiez les niveaux logiques et limitez la tension appliquée sur l'entrée série du « GM862 » à 3,3 V afin d'éviter sa destruction. Dans tous les cas, on éloignera le plus possible le module GSM et surtout son antenne du CUBLOC et de sa platine de test « CUBLOC Study Board » afin d'éviter que des retour « HF » ne viennent perturber ces derniers.



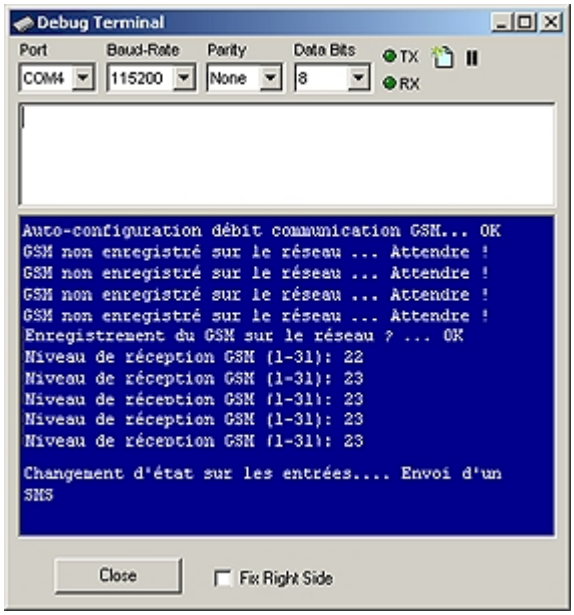
Saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « gsm1 »).

Principe de fonctionnement de l'application :
Le programme consiste à envoyer des commandes « AT » reconnues par le module « GM862 » afin de pouvoir lui permettre d'envoyer des « SMS » d'alertes en cas de modification de la position des 2 interrupteurs.

Note : Comme explicitement indiqué dans le titre de cette note d'application, la description de cette réalisation se veut ludique et doit être utilisée uniquement à titre expérimental. Cette note d'application ne doit jamais être utilisée telle quelle dans le cadre de la réalisation d'un système d'alerte pour quelque application que ce soit et à fortiori pour la réalisation d'une application qui pourrait mettre en périls des personnes, des animaux ou des biens matériels en cas de défaillance de celle-ci. LEXTRONIC décline toute responsabilité en cas de non-respect de cette mise en garde.

La raison principale de cette mise en garde vient du fait que s'agissant d'une simple description ludique et afin de ne pas surcharger le programme, l'application présentée repose sur une utilisation « simplifiée » du module « GM862 » dans laquelle nous avons volontairement fait l'impasse sur plusieurs tests normalement nécessaires pour s'assurer pleinement de la validité de la configuration de la communication du module « GM862 » (ces possibilités de tests sont décrites en détail dans la notice du module « GSM »).

De plus aucune vérification sur la validité de la connexion série et la validité des données en provenance du module « GM862 » n'est effectuée.



Le programme débute par la configuration de 3 broches du CUBLOC en entrée et par la configuration de la vitesse de communication de son port série à 38400 bds.

S'en suit alors une boucle sans fin dans laquelle le CUBLOC va tenter de configurer la vitesse de communication du module « GM862 » en lui envoyant les caractères ASCII « AT » et en attendant en retour la chaîne de caractères « OK ». Cette chaîne est extraite du buffer de réception série du CUBLOC (dont il en analyse un emplacement précis – celui où doit figurer les caractères OK). La boucle sans fin est continuellement réalisée tant que le CUBLOC n'obtient pas satisfaction. Si les caractères « OK » sont détectés, il en résulte que le module « GM862 » a bien détecter la requette du « CB220 » et qu'il sera désormais prêt à communiquer avec lui à 38400 bds (pour l'occasion, le « CB220 » affiche alors un petit message de confirmation (« Auto-configuration débit communication GSM … OK ») dans la fenêtre de Debug et sort de la boucle sans fin.

Une seconde boucle sans fin sera alors effectuée au cours de laquelle le « CB220 » demande au module « GM862 » de lui confirmer qu'il s'est bien connecté au réseau « GSM » en lui envoyant une requête sous la forme de la chaîne ASCII « AT+CREG ». Le « CB220 » attendra une réponse « positive » devant se présenter sous la forme d'une chaîne de caractères « +CREG: 0,1 » ou « +CREG: 0,1 » (consultez la doc. Du « GM862 » pour plus d'infos). Si telle n'est pas le cas, le « CB220 » affichera à chaque tentative un message adéquat à sur la fenêtre de Debug du PC « GSM non enregistré sur le réseau… Attendre ! ». Cette opération peut prendre plusieurs dizaines de secondes ou de minutes suivant l'état du réseau et l'endroit où vous vous trouvez. Dès que la connexion est établie, un nouveau message « Enregistrement du GSM sur le réseau ? ….. OK» s'affiche et le « CB220 » sort de la boucle sans fin.

A ce stade, le « CB220 » envoi une commande spéciale au « GM862 » afin de le configurer en mode « SMS / TEXT ». Le « CB220 » va ensuite vérifier et mémoirser l'état initial de ses 2 entrées tout ou rien.

Dès lors, le module « CB220 » réalise une troisième boucle sans fin au cours de laquelle il va s'assurer de la viabilité du niveau de réception (force du signal) du module « GM862 ». Si ce dernier n'est pas suffisant, il vous affichera un message adéquat dans la fenêtre de Debug du PC. En revanche, si le niveau est exploitable, il vous affichera (toujours dans la fenêtre du PC) la force du signal sous la forme d'un chiffre compris entre 6 et 31 (plus le chiffre est élevé – meilleur sera la réception) et il sortira de cette boucle sans fin pour vérifier l'état des entrées « tout ou rien ».

Si aucune modification sur l'état logique de ses entrées n'est intervenues, le « CB220 » recommence un test de niveau de réception et ainsi de suite.

Si en revanche une ou plusieurs entrées ont changé d'état, le « CB220 » envoi une commande au module GSM pour lui demander d'envoyer un « SMS » à un numéro de téléphone (les xxxxxxxx suivant le 06 sur le programme sont à remplacer par la fin du N° de téléphone que vous voulez appeler). Le « CB220 » va ensuite envoyer une suite de caractères ASCII correspondant au message à envoyer. Ainsi suivant l'état des entrées tout-ou-rien il enverra des messages indiquant par exemple « Entrée 1 : fermée » ou « Entrée 2 : fermée ».

Le « CB220 » récupérera également le résultat de la conversion « A/N » de sa première entrée afin de composer également un message donnant (grâce à une petite conversion) la valeur de la tension présente sur cette entrée. La composition du message « SMS » se traduit par l'envoi du caractère « &h01a ».

Le programme re-mémorise alors à nouveau le niveau logique des entrées « tout ou rien » avant d'envoyer une commande « ESC » au modem et de recommencer toute l'exécution depuis le début via l'instruction « RESET » !

```
'####################################
'#        Gestion d'un module GSM          #
'#                  « GM-862 »                 #
'#     @Lextronic 2006 - 09/10/2006        #
'####################################

Const Device = CB220

Dim reponse As String * 35
Dim i As Byte
Dim e1 As Byte
Dim e2 As Byte
Dim tension As Single

Input 0                                    ' Configure port 0 en entree (pour utilisation en analogique)
Input 1                                    ' Configure port 1 en entree
Input 2                                    ' Configure port 2 en entree
Opencom 1,38400,3,100,100

Delay 3000

' ---------- Auto-configuration vitesse communication avec le GSM  ----------

Do
    Bclr 1,0                               ' RAZ buffer réception
    Delay 800
    Putstr 1,"AT",13,10                    ' Envoi commande "AT"
    Delay 800                               ' Tempo pour laisser le temps au GSM de répondre
    reponse = Getstr(1,10)                 ' Récupère réponse du GSM
    If Mid(reponse,7,2)="OK" Then Exit Do  ' Regarde si le GSM s'est bien configuré ?
   Loop
   Debug "Auto-configuration débit communication GSM... OK",13,10

  ' ---------- Test enregistrement du GSM sur le réseau  ----------

   Do
    Putstr 1,"AT+CREG",13,10               ' Envoi commande "AT+CREG"
    Delay 800                               ' Tempo pour laisser le temps au GSM de répondre
    reponse = Getstr(1,29)                 ' Récupère réponse du GSM
    If Mid(reponse,12,10)="+CREG: 0,1" Or Mid(reponse,12,10)="+CREG: 1,1" Then
      Debug "Enregistrement du GSM sur le réseau ? ... OK",Cr
      Exit Do
    Else
     Debug "GSM non enregistré sur le réseau ... Attendre !",Cr
    Endif
   Loop

   ' ---------- Configure SMS en mode "Text"  ----------

   Putstr 1,"AT+CMGF=1",13,10              ' Envoi commande "AT+CREG"
```

```
Delay 800
Bclr 1,0                                    ' RAZ buffer réception

' ------------------ Boucle principale / Test et afiche niveau de réception signal GSM  ----------------------------

e1 = In(1)                                  ' Mémorise état initial entrée E1
e2 = In(2)                                  ' Mémorise état initial entrée E2
Do
  Do
   Putstr 1,"AT+CSQ",13,10                  ‘ Envoie commande "AT+CSQ"
   Delay 800                                ' Tempo pour laisser le temps au GSM de répondre
   reponse = Getstr(1,26)                   ' Récupère réponse du GSM
   Bclr 1,0                                 ' RAZ buffer réception
   i = 1                                    ' Initialise position dans la chaîne de caractères reçus
   Do
     If Mid(reponse,i,1)="," Then Exit Do   ' Recherche la virgule de séparation
     i = i+1                                ' Se position au caractère suivant
   Loop
   reponse = Mid(reponse,i-2,2)
   If Val(reponse)<6 Or Val(reponse)= 99 Then
     Debug "Niveau de réception GSM insuffisant...  !",Cr
   Else
     Debug "Niveau de réception GSM (1-31): ",Dec(Val(reponse)),Cr
     Exit Do
   Endif
  Loop

  '-------------------- Vérification d'un changement sur les entrées ------------------------------------------

  If e1 <> In(1) Or e2 <> In(2) Then          ' Test si modification état des entrées
    Debug Cr,"Changement d'état sur les entrées.... Envoi d'un SMS !",Cr
    e1 = In(1)                              ' Mémorise nouvel état entrée E1
    e2 = In(2)                              ' Mémorise nouvel état entrée E2
    tension=(5.0/1023.0)*Adin(0)            ' Mémorise valeur de la tension du potentiomètre
    reponse=Left(Float(tension),4)
    Putstr 1,"AT+CMGS=06xxxxxxxx",13,10' Envoi commande pour SMS vers ce N° de téléphone
    Delay 1500
    Putstr 1,"Alarme CUBLOC !",13,10,13,10
    If e1 = 1 Then
      Putstr 1,"Entrée 1: ouverte",13,10
    Else
      Putstr 1,"Entrée 1: fermée",13,10
    Endif
    If e2 = 1 Then
     Putstr 1,"Entrée 2: ouverte",13,10
    Else
     Putstr 1,"Entrée 2: fermée",13,10
    Endif
    Putstr 1,"Tension: ",reponse," Vcc",13,10,&h01A
    Delay 9000
    Putstr 1,&h01B                          ' Caractère ESC
    Reset                                   ' Recommence tout le programme depuis le début !
  Endif
Loop
```

NOTE D'APPLICATION # 39.
Lecture fichier MP3 via platine « uMP3 »

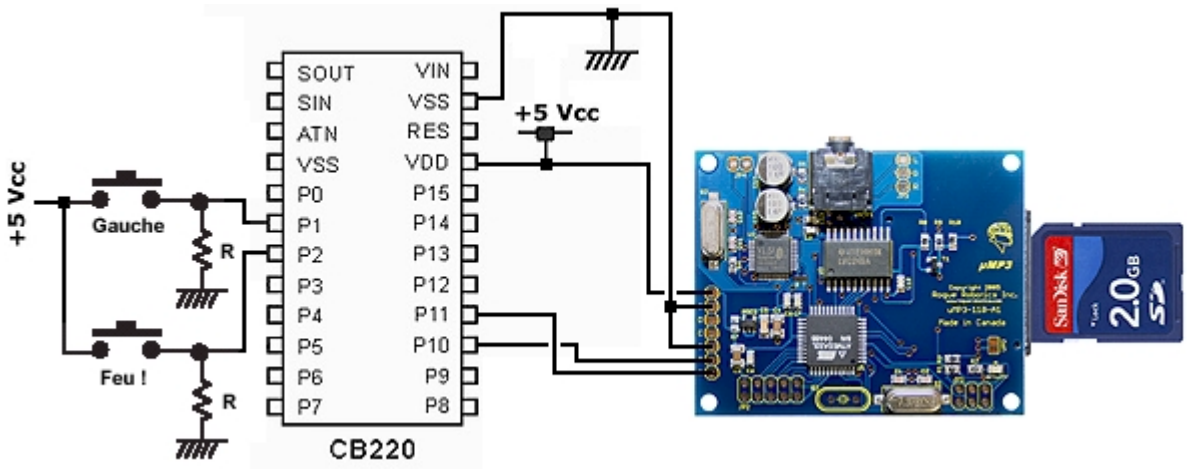
Cette note d'application va vous permettre de piloter une platine « uMP3 » afin de pouvoir restituer des enregistrements sonores à partir des fichiers "MP3" préalablement stockés par vos soins sur une carte SD™/ MMC™ (**non livrée**). La platine se pilote au moyen d'une liaison série 9600 bds. Nous utiliserons uniquement que quelques commandes de base pour afficher la version logiciel de la platine, afficher le contenu de la carte SD™ dans la fenêtre de debug du PC ou encore utiliser les commandes de lecture / pause / stop de la platine.

Notions abordées :
- Gestion communication série

Matériel nécessaire :
- Un module « CUBLOC CB220 »
- Une platine « uMP3 »
- Une carte SD™ (sur laquelle vous devrez préalablement avoir stockés des fichiers MP3)
- Un écouteur.

Préparation matérielle :

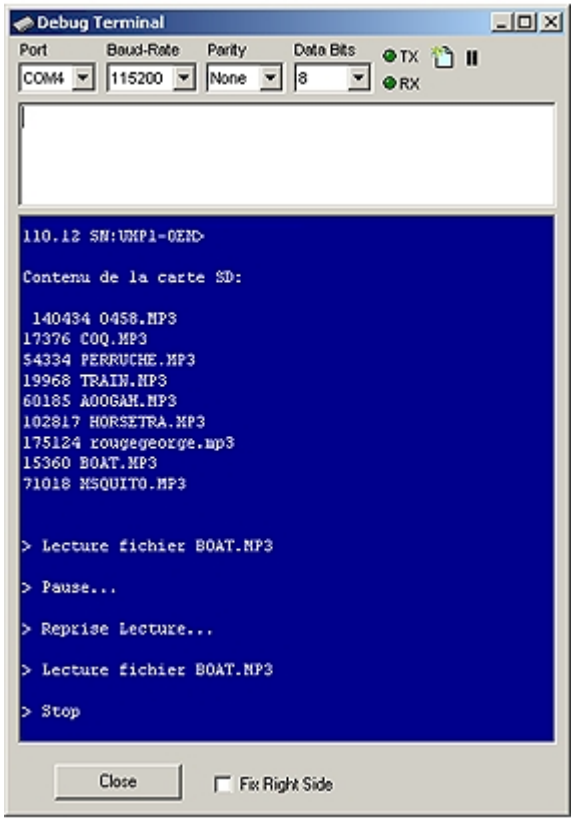
Celle-ci repose en fait sur un simple raccordement entre le port série du « CB220 » et celui de la platine « uMP3 ». On utilisera également 2 boutons-poussoirs sur les entrées du « CB220 » pour commander les fonctions de « Lecture / Pause / Stop ».



Saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « ump3 »).

Principe de fonctionnement de l'application :
Le programme est très simple et consiste à envoyer des commandes séries au module « uMP3 ». La première commande demande au module « uMP3 » de lui retourner le N° de version de son Firmware (le « CB220 » affiche cette information dans la fenêtre de DEBUG du PC).

La seconde commande demande au module « uMP3 » de lui retourner la liste des fichiers présents sur la carte SD™ (il faudra bien évidemment que vous ayez préalablement transféré des fichiers MP3 sur cette carte avant de l'insérer dans le module « uMP3 ». Le Cubloc va après l'envoi de cette commande afficher les caractères un à un sur la fenêtre de DEBUG du PC. Des tests sont effectués pour détecter les caractères de « saut de ligne » ainsi que le caractère indiquant la fin de la liste (>) qui permettra de sortir de la boucle sans fin de lecture du buffer série (via l'instruction « exit do »). A ce stade, la fenêtre DEBUG du PC vous affichera successivement la taille mémoire et le nom des fichiers présents sur la carte SD™.



Le reste du programme est une boucle sans fin dans laquelle on test la sollicitation de 2 entrées du module CUBLOC. Si la première entrée est sollicitée, le « CB220 » enverra en premier lieu une commande au module «uMP3 » afin de connaître son état.

Si la première lettre à gauche de la chaîne de caractère reçue en réponse de la part du module « uMP3 » est un « S », le « CB220 » en déduit que le module « uMP3 » n'est pas en cours de restitution de fichier et le CUBLOC pourra alors envoyer une commande de lecture de fichier MP3 (pour cet exemple on ne choisira de lire qu'un seul fichier « BOAT.MP3 » en affichant un message dans la fenêtre de débug).

Si à ce stade, on sollicite à nouveau la première entrée, le « CB220 » après interrogation du module « uMP3 » recevra alors une chaîne de caractère dont la première lettre lui indiquera que le module est en cours de lecture d'un fichier, le « CB220 » enverra alors une commande de pause puis suivant la nature de la première lettre de la chaîne de caractère reçue, on affichera un message indiquant qu'on a effectué une commande de pause ou de reprise de la lecture du fichier (en effet, à chaque envoi d'une commande de pause on vient mettre le fichier en cours de lecture en pause ou en reprise de lecture.

La première touche du « CB220 » servira donc à la fois à initier la lecture du fichier MP3, mais aussi à effectuer des pauses ou des reprises de lecture à chacune de sa sollicitation.

La seconde touche du « CB220 » permet quand à elle s'envoyer une commande de « Stop » du fichier en cours de lecture. Il n'est pas possible dans ce cas de reprendre la lecture du message à l'endroit où on l'a stoppé comme cela se fait avec la commande de pause.

Comme indiqué au début de cette note d'application, il est possible de réaliser bien d'autres fonctions avec cette platine « uMP3 ». Vous pourrez ainsi très facilement et selon le même principe d'envois de commandes séries : modifier le volume sonore, connaître la position de restitution d'un fichier en cours de lecture, mais également lire et écrire dans des fichiers de la carte SD™ afin de l'utiliser comme unité de sauvegarde, etc, etc… Libre à vous de découvrir et d'exploiter ces possibilités pour les besoins de vos applications.

```
'###################################
'#        Gestion d'un module                #
'#                « uMP3 »                   #
'#     @Lextronic 2006 - 22/10/2006       #
'###################################

Const Device = CB220

Dim reponse As String * 26
Input 1                                         ' Configure port 1 en entree
Input 2                                         ' Configure port 2 en entree

Opencom 1,9600,3,100,100

Delay 3000

' ---------- Récupère et affiche la version du logiciel  ----------

Bclr 1,0                                        ' RAZ buffer réception
Putstr 1,"V",13,10                              ' Envoi commande pour connaître la version logiciel du module
Delay 500                                       ' Tempo pour laisser le temps au module de répondre
reponse = Getstr(1,25)                          ' Récupère la version du logiciel du module "uMP3"
Debug reponse,Cr,Cr                             ' Affiche la version du logiciel

' ---------- Récupère et affiche contenu de la carte SD  ----------

Debug "Contenu de la carte SD:",Cr,Cr
Bclr 1,0                                        ' RAZ buffer réception
```

```
Putstr 1,"FC L/",13,10                          ' Envoi commande pour connaître le contenu e la carte SD
Do
   reponse = Getstr(1,1)                        ' Récupère caractère par caractère
   If reponse = ">" Then Exit Do                ' Regarde si fin de la liste ?
   If Asc(reponse) = 13 Then Debug Cr           ' Regarde si on saute une ligne ?
   Debug reponse                                ' Affiche les caractères reçus
Loop
Debug Cr,Cr

'---------------- Restitution des messages MP3 -----------------------------------

Do
  If In(1)=0 Then
  Bclr 1,0                                      ' RAZ buffer réception
  Putstr 1,"PC Z",13,10                         ' Envoi commande pour connaître l etat du module
  Delay 500                                     ' Tempo pour laisser le temps au module de répondre
  reponse = Getstr(1,25)                        ' Récupère l etat du module "uMP3"
  If Left(reponse,1)="S" Then                   ' Regarde si le module etait a l arret ?
     Putstr 1,"PC F/BOAT.MP3",13,10             ' Envoi commande de lecture du fichier "BOAT"
     Pause 500
     Debug "> Lecture fichier BOAT.MP3",Cr,Cr
  Else
    Putstr 1,"PC P",13,10                       ' Envoi commande de Pause
    If Left(reponse,1)="D" Then                 ' Regarde si le module etait en pause ?
       Debug "> Reprise Lecture...",Cr,Cr
    Else
       Debug "> Pause...",Cr,Cr
    Endif
  Endif
 Endif

 If In(2)=0 Then
  Putstr 1,"PC S",13,10                         ' Envoi commande de stop
  Pause 500
  Debug "> Stop",Cr,Cr
 Endif
Loop
```

NOTE D'APPLICATION # 40.
Pilotez un robot « marcheur »

Cette note d'application va vous permettre de piloter un châssis de robot marcheur. Livrée en kit et sans électronique, la base robotique « Troll » vous permettra en association avec 2 servo-moteurs (livrés en option) de réaliser un robot "marcheur" bipède extrêmement ludique. Le premier servomoteur servira à soulever l'une ou l'autre des jambes du robot tandis que le second servo-moteur servira à faire avancer le robot.

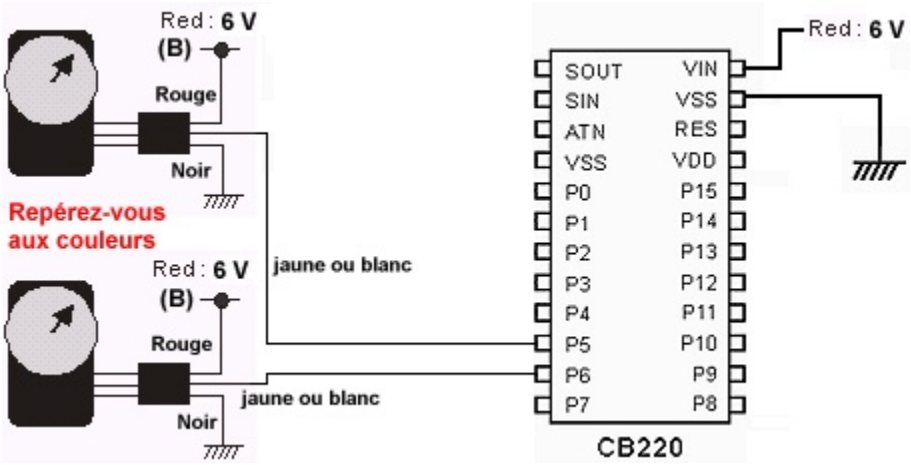
Nous utiliserons une platine « PROTO-220 » associée à un CB220 pour piloter les 2 servomoteurs.

Notions abordées :
- Gestion de servomoteurs

Matériel nécessaire :
- Une platine « CB220-PROTO »
- Un module « CB220 »
- Une base robotique « Troll »
- 2 Servomoteurs + 1 support pour 4 piles 1,5 V + 4 piles 1,5 V

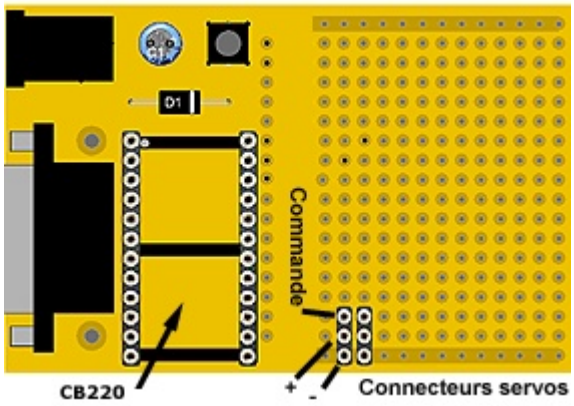
Préparation matérielle :

Celle-ci repose en fait sur un simple raccordement entre 2 ports PWM du « CB220 » et les entrées de commande des servomoteurs.

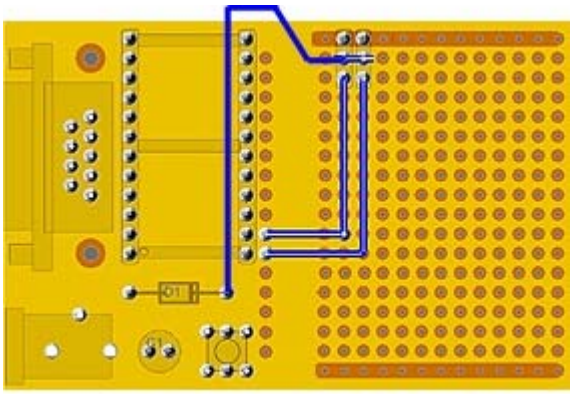


On utilisera 4 piles de 1,5 V (soit 6 Vcc) pour alimenter à la fois le « CB220 » (attention il vous faudra utiliser l'entrée VIN et non VDD), mais aussi l'alimentation des servomoteurs.

Pour faciliter la réalisation on pourra avoir recours à une platine « CB220-PROTO » sur laquelle on effectuera les câblages ci-dessous (on remarquera le « strap » permettant de ramener la tension d'alimentation + 6 Vcc sur l'alimentation des servomoteurs. Attention lors de la connexion des servomoteurs à ne pas inverser le sens des connecteurs de ces derniers sous peine de destruction.



Platine « CB220-PROTO » vue de dessus



Platine « CB220-PROTO » vue de dessous

Saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « troll »).

Principe de fonctionnement de l'application :
Le programme est très simple et consiste à piloter les 2 servomoteurs pour faire avancer le robot en décomposant les différents mouvements.

La première opération consiste à positionner les palonniers des servomoteurs au neutre afin que le robot soit en position de départ.

Ensuite le programme change lentement la position du servomoteur N° 1 afin de lever le pied gauche (une tempo delay 1 est réalisée entre chaque modification de position du servomoteur).

Une fois le pied levé, on modifie la position du deuxième servomoteur pour avancer le pied gauche en avant.

Puis on baisse le pied gauche et on lève le pied droit pour ensuite avancer ce dernier avant de le baisser et de lever l'autre pied pour l'avancer et ainsi de suite… (les mouvements s'enchaînant dans une boucle sans fin).

Vous disposez d'une petite vidéo sur notre site pour vous montrer le résultat.

A noter que nous avons eu recours à la réalisation d'une petite routine de tempo car l'utilisation de l'instruction délai rendait le déplacement trop lent.

Dans le cadre de votre réalisation, il vous faudra probablement modifier les valeurs des signaux PWM donnés dans notre programme afin d'adapter les mouvements à vos servomoteurs (chaque servo ayant des tolérances différentes).

Dans tous les cas, évitez les mouvements brusques sur les servomoteurs afin de ne pas déséquilibrer le châssis du robot et provoquer sa chute.

Le bloc de pile pourra être logé dans le corps du robot. Si vous réalisez vous-même votre platine de commande, il vous sera possible alors de loger l'ensemble de l'électronique ainsi que les pile dans le robot afin que rien ne soit apparent de l'extérieur (pour notre part, nous avons placé la platine de commande « CB220-PROTO » à l'extérieur.

En vous aidant des notes d'applications déjà proposées, vous pourrez également aisément ajouter des capteurs capables d'indiquer la présence d'obstacles au robot afin qu'il les évite !

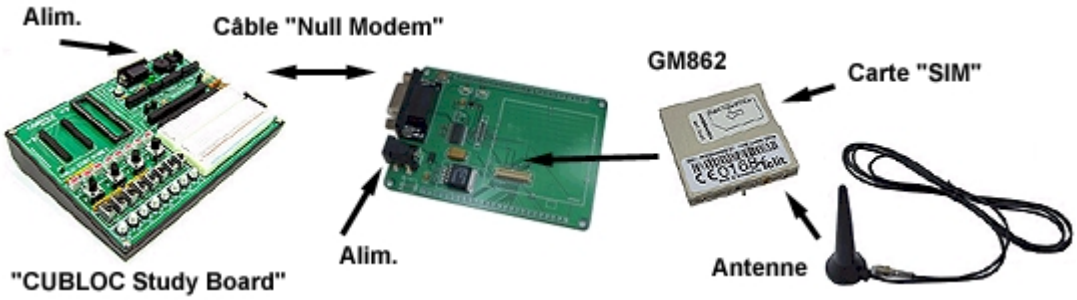
```
'###################################
'#        Gestion d'une base de robot        #
'#                  « marcheur »                #
'#     @Lextronic 2006 - 25/10/2006         #
'###################################

Const Device = CB220

Dim position As Integer
Dim A As Byte

Low 5                                   ' Configure les port P5 et P6 en sortie
Low 6

Pwm 0,3230,32768                        ' Robot en position normale
Pwm 1,3109,32768

'--------- Départ position initiale ----------------------------------

For position = 3230 To 2850 Step-1       ' Lève pied gauche
  Pwm 0,position,32768
  Delay 1
Next

For position = 3109 To 3700              ' Avance pied gauche
  Pwm 1,position,32768
  Gosub tempo
Next

'--------- boucle principale marche ----------------------------------

Do
  For position = 2850 To 3580            ' Repose pied gauche et lève pied droit
    Pwm 0,position,32768
    Delay 1
  Next

  For position = 3700 To 2590 Step-1     ' Avance pied droit
    Pwm 1,position,32768
    Gosub tempo
  Next

  For position = 3580 To 2850 Step-1     ' Repose pied droit et lève pied gauche
    Pwm 0,position,32768
    Delay 1
  Next

  For position = 2590 To 3700            ' Avance pied gauche
    Pwm 1,position,32768
    Gosub tempo
  Next
Loop

'---------- Routine de temporisation ----------------------------------
tempo:
   For A = 1 To 3
   Next
   Return
```

NOTE D'APPLICATION # 41.
Télécommande « GSM » expérimentale

Cette note d'application va vous permettre de réaliser une télé-comande « GSM » expérimental. Pour ce faire nous utiliserons un module GSM/GPRS « GM862 » "Quad-band" capable de supporter les modes de communication "voix", "fax", "Data", "SMS". Ce dernier se caractérise par ses faibles dimensions associées à un prix très compétitif et à une grande simplicité de mise en oeuvre. L'application permettra à un module CUBLOC « CB220 » de piloter 2 sorties « tout-ou-rien » à distance via l'envoi de « SMS » et de recevoir en retour un autre « SMS » qui vous confirmera l'état des sorties et ainsi la bonne prise en compte de votre commande.

Afin de simplifier la réalisation, on aura recours à une platine de test spécialement conçue pour pouvoir recevoir le module GSM « GM862 ». Cette platine comprend un étage de régulation ainsi qu'un étage de mise en forme du signal de sortie du module « GSM » (afin que les niveaux logiques de celui-ci soit compati-bles avec une liaison RS232. Il nous suffira alors de connecter une alimentation à cette platine, une antenne au module GSM et d'utiliser un « NULL-MODEM » pour relier cette carte à la platine de test « CUBLOC Study Board » via leur port série.

La dernière opération indispensable consistera à se munir bien évidemment d'une carte « SIM » opérationnelle que l'on aura inséré (hors alimentation) dans le module « GM862 » grâce au connecteur prévu à cet effet. Le code « PIN » de la carte SIM ainsi que tous les « SMS » en mémoire devront être effacés afin de simplifier la procédure d'utilisation du module GSM.



On utilisera également de préférence 2 alimentations séparées (une pour la platine du GSM et une pour la platine « PB Study Board ».

Egalement libre à vous de réaliser directement votre propre platine support pour le module « GM862 ».

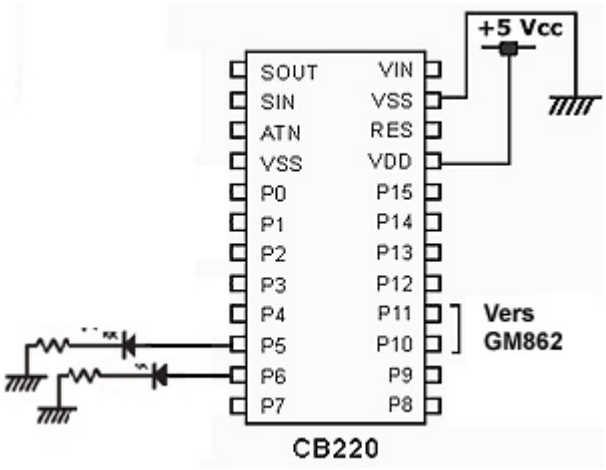
Notions abordées :
- Gestion communication série

Matériel nécessaire :
- Un module « CUBLOC CB220 »
- Une platine « PB Study Board »
- Un module « GM862 »
- Une antenne GSM
- Une platine d'interface pour « GM862 »
- 2 blocs d'alimentation
- 1 câble « Null Modem »
- 1 carte SIM opérationnelle (**avec code PIN effacé**).

Préparation matérielle :

Celle-ci repose en fait sur un simple raccordement entre le port série du « CB220 » et celui du module « GM862 » et au câblage de 2 ports du « CB220 » sur des Leds.

Si vous ne passez pas par les platines de tests évoquées ci-avant et que vous reliez directement le module « GSM » au CUBLOC, vérifiez les niveaux logiques et limitez la tension appliquée sur l'entrée série du « GM862 » à 3,3 V afin d'éviter sa destruction. Dans tous les cas, on éloignera le plus possible le module GSM et surtout son antenne du CUBLOC et de sa platine de test « CUBLOC Study Board » afin d'éviter que des retour « HF » ne viennent perturber ces derniers.



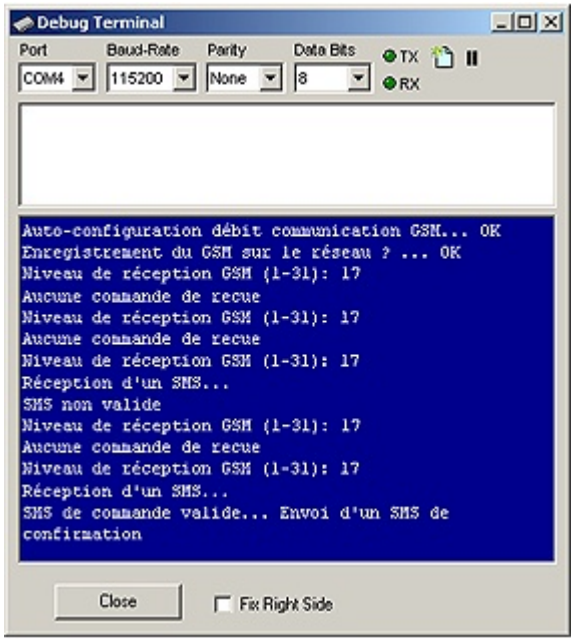
Saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « gsm2 »).

Principe de fonctionnement de l'application :
Le programme consiste à envoyer des commandes « AT » reconnues par le module « GM862 » afin de pouvoir lui permettre de vérifier la réception de message « SMS » afin d'en analyser le contenu et de piloter si nécessaire 2 sorties du CUBLOC et de renvoyer un « SMS » de confirmation (accusé de reception).

Note : Comme explicitement indiqué dans le titre de cette note d'application, la description de cette réalisation se veut ludique et doit être utilisée uniquement à titre expérimental. Cette note d'application ne doit jamais être utilisée telle quelle dans le cadre de la réalisation d'une application qui pourrait mettre en périls des personnes, des animaux ou des biens matériels en cas de défaillance de celle-ci. LEXTRONIC décline toute responsabilité en cas de non-respect de cette mise en garde.

La raison principale de cette mise en garde vient du fait que s'agissant d'une simple description ludique et afin de ne pas surcharger le programme, l'application présentée repose sur une utilisation « simplifiée » du module « GM862 » dans laquelle nous avons volontairement fait l'impasse sur plusieurs tests normalement nécessaires pour s'assurer pleinement de la validité de la configuration de la communication du module « GM862 » (ces possibilités de tests sont décrites en détail dans la notice du module « GSM »).

De plus aucune vérification sur la validité de la connexion série et la validité des données en provenance du module « GM862 » n'est effectuée. Enfin et surtout sachez que le programme donné en exemple effacera systématiquement tous les « SMS entrant » après avoir essayer de déterminer s'il s'agit d'un « SMS » de commande. Donc en utilisant ce programme, vous pourrez « louper » des « SMS » qui pourront peut être vous avoir été envoyé au moment où vous mettez l'application en service. De plus, le programme ne gère que les cas de figure de réception d'un « SMS » à la fois (le programme ne vérifie que l'emplacement mémoire N° des « SMS » car encore une fois, il ne s'agit que d'un programme d'expérimentation).



Le programme débute par la même « structure » et procédure d'initialisation que celle décrite dans la note d'application N° #38 de ce même document. Cette dernière consiste à initialiser le débit de communication avec le « GSM », à s'assurer que le « GSM » est bien connecté au reseau et à configurer les SMS en mode « Texte » (voir les explications de la note d'application N°# 38 pour plus d'infos).

Toutefois la distinction principale est que le « CB220 » envoi également une commande « AT » au module « GSM » afin d'effacer d'office le message « SMS » potentiellement présent à l'adresse mémoire 1. Pensez donc à consulter vos messages « SMS » avant de mettre en œuvre cette application.

La boucle principale du programme début par l'affichage du niveau de réception (force du signal) et le cas échéant par l'affichage d'un message d'alerte dans la fenêtre de DEBUG du PC si ce niveau est insuffisant. Le programme envoi ensuite une commande « AT » afin de récupérer le contenu du « SMS » présent à l'emplacement mémoire N° 1. Si aucun message « SMS » n'est arrivé, le module « GSM » envoi une chaîne de caractères comportant (entre autre) le mot ERROR que le « CB220 » va essayer de détecter. Si cette chaîne de caractères est bien détectée, le « CB220 » indique dans la fenêtre de DEBUG qu'aucun message « SMS » n'est arrivé, puis le niveau de réception est de nouveau affiché et ainsi de suite…

Pour piloter à distance les 2 sorties du « CB220 », il vous faudra vous munir d'un second téléphone portable et envoyer un « SMS » vers le N° de la carte SIM que vous avez inséré dans le module « GM862 ». Ce « SMS » devra être du style :

s1=on ou **s2=on** ou **s2 =on** ou **s2=on**

Toutes les lettres doient être en minuscule et tous les caractères devront être collés (il ne doit pas y avoir d'espace). Par exemple pour activer la sortie 1, il vous faudra envoyer le message : **s1=on** et pour désactiver la sortie 2, il vous faudra envoyer le message : **s2=off**

Dès lors, en cas de réception d'un « SMS », le « CB220 » va analyser la chaîne de caractères présente dans la mémoire N° 1 du « SMS » et essayer de détecter les caractères **s1=on** ou **s2=on** ou **s2 =on** ou **s2=on** .Si ces caractères ne sont pas reconnus, le « CB220 » considère que le « SMS » reçu n'est pas valide et ne prend pas en compte les modifications de ses sorties et affiche un message indiquant dans la fenêtre de DEBUG que le « SMS » n'est pas valide, puis il retourne en attente d'un nouveau « SMS » (en ayant pris soin d'effacer le « SMS » non reconnu).

Si en revanche les chaînes de caractères **s1=on** ou **s2=on** ou **s2 =on** ou **s2=on** sont reconnues, le « CB220 » modifiera « l'image » des sorties mémorisée dans les variables « SORTIE1 et SORTIE2 » avant de modifier l'état physique de ces sorties pour de bon sur ses ports. Le message « SMS » ainsi reçu et « décodé » sera alors effacé de l'emplacement mémoire N°1 et la boucle principale du programme se terminera enfin par l'envoi d'un « SMS » de confirmation qui indiquera l'état des sorties via un message « en clair » composé en fonction des valeurs des variables « SORTIE1 et SORTIE2 ». Lors de la composition de ce message, vous devrez dans le programme indiqué au « GSM » le n° de téléphone « GSM » à qui vous devrez envoyer le « SMS » (ce N° est matérialisé par la chaîne 06xxxxxxxx dans le programme ou les xxxxxxxx seront à remplacer par la suite de vos propres n°). une fois le « SMS » envoyé, le programme reprend sa boucle initiale en attente d'un nouveau « SMS ».

A noter que le décodage des « SMS » est réalisé de façon sommaire (en effet n'importe quel autre utilisateur qui vous enverra un « sms » correctement « formaté » du type **s1=on** ou **s2=on** ou **s2 =on** ou **s2=on** pourra activer ou désactiver les sorties du « CB220 » (la seule limitation est qu'il ne recevra pas de « SMS » d'accusé de réception si vous n'avez pas programmé son N° de téléphone dans la génération du « SMS » de retour. Afin de palier à cette limitation, vous avez 2 solutions : Soit vous choisissez un « SMS » plus sophistiqué avec par exemple un code devant la commande (ex : **1457856s1=on** - ici vous devrez alors modifier le programme pour vous assurer que le « CB220 » vérifie la présence du code 1457856) ou alors vous pourrez également vérifier la présence et la validité du n° de téléphone étant à l'origine du « SMS ». En effet comme vous pourrez le constater en consultant la documentation du « GM862 », ce dernier vous restitue le N° de téléphone à l'origine des « SMS » entrant. Libre à vous de vérifier si le n° de téléphone est autorisé ou non à piloter votre « CB220 ».

Dernier point à souligner, il est également conseillé au lieu de travailler avec les variables mémoires SORTIE1 et SORTIE2 (qui sont le reflet des sorties physiques du « CB220 ») d'utiliser 2 relais (à 2 contacts RTC) et à double bobines avec effet mémoire pour gérer les signaux de sorties du « CB220 ». Chaque relais devra être piloté (via 2 étages à transistor) par 2 sorties du « CB220 ». Sous l'action de la première sortie, la première bobine du relais fera « coller » ce dernier en conservant cet état grâce à un effet « mémoire magnétique » (c'est à dire qu'une simple impulsion sur la sortie du « CB220 » suffira à maintenir la position collée du relais même en cas de coupure d'alimentation. Une nouvelle impulsion sur la même bobine ne modifiera pas l'état du relais. Par contre, une impulsion sur la seconde bobine (donc via la seconde sortie du « CB220 ») provoquera le décollage du relais. Cette méthode nécessite donc 4 sorties au niveau du « CB220 » pour piloter 2 relais. Toutefois l'avantage est que vous pourrez utiliser le premier contact du relais pour votre application et le second contact du relais pour connaître son état (via une entrée supplémentaire à utiliser au niveau du « CB220 » - le recours à cette methode nécessite donc l'utilisation en tout de 6 broches du « CB220 » pour exploiter 2 relais bistables). Le fait de connaître l'état « physique » du relais (fermé ou ouvert) via une entrée logique au détriment d'une gestion « logique » de l'état des sorties via des variables mémoire en RAM est de bénéficier d'une meilleure sécurité en cas de coupure d'alimentation ou d'une réinitialisation du programme. En effet dès le début du programme, on force les variables SORTIE1 et SORTIE2 à 0, ce qui peut fausser l'idée de la position que l'on pensait avoir envoyé si une réinitialisation du programme du « CB220 » intervient pour une raison ou pour une autre.

```
'###################################
'#        Gestion d'un module GSM          #
'#                « GM-862 »                 #
'#     @Lextronic 2006 - 25/10/2006        #
'###################################

Const Device = CB220

Dim reponse As String * 35
Dim i As Byte
Dim sortie1 As Byte
Dim sortie2 As Byte

Low 5                                                   ' Configure port 5 en sortie et force niveau à 0
Low 6                                                   ' Configure port 6 en sortie et force niveau à 0

Opencom 1,38400,3,100,100

sortie1 = 0
sortie2 = 0
Delay 3000

' ---------- Auto-configuration vitesse communication avec le GSM  ----------

Do
   Bclr 1,0                                             ' RAZ buffer réception
   Delay 800
   Putstr 1,"AT",13,10                                  ' Envoi commande "AT"
   Delay 800                                            ' Tempo pour laisser le temps au GSM de répondre
   reponse = Getstr(1,10)                               ' Récupère réponse du GSM
   If Mid(reponse,7,2)="OK" Then Exit Do                ' Regarde si le GSM s'est bien configuré ?
 Loop
 Debug "Auto-configuration débit communication GSM... OK",13,10

 ' ---------- Test enregistrement du GSM sur le réseau  ----------

 Do
    Putstr 1,"AT+CREG",13,10                            ' Envoi commande "AT+CREG"
    Delay 800                                           ' Tempo pour laisser le temps au GSM de répondre
    reponse = Getstr(1,29)                              ' Récupère réponse du GSM
    If Mid(reponse,12,10)="+CREG: 0,1" Or Mid(reponse,12,10)="+CREG: 1,1" Then
      Debug "Enregistrement du GSM sur le réseau ? ... OK",Cr
      Exit Do
    Else
     Debug "GSM non enregistré sur le réseau ... Attendre !",Cr
    Endif
  Loop

 ' ---------- Configure SMS en mode "Text"  ----------

 Putstr 1,"AT+CMGF=1",13,10                             ' Envoi commande "AT+CREG"
 Delay 800
 Bclr 1,0                                               ' RAZ buffer réception

 ' ---------- Efface le SMS en mémoire en position N° 1  ----------

 Putstr 1,"AT+CMGD=1",13,10                             ' Envoi commande "AT+CMGD"
 Delay 2000
 Bclr 1,0                                               ' RAZ buffer réception

 ' ---------- Boucle principale / Test et affiche niveau de réception signal GSM  ----------

   Do
     Do
```

```
 Bclr 1,0                                          ' RAZ buffer réception
 Putstr 1,"AT+CSQ",13,10                           ' Envoie commande "AT+CSQ"
 Delay 800                                         ' Tempo pour laisser le temps au GSM de répondre
 reponse = Getstr(1,26)                            ' Récupère réponse du GSM
 Bclr 1,0                                          ' RAZ buffer réception
 i = 1                                             ' Initialise position dans la chaîne de caractères reçus
 Do
  If Mid(reponse,i,1)="," Then Exit Do             ' Recherche la virgule de séparation
  i = i+1                                          ' Se positionne au caractère suivant
 Loop
 reponse = Mid(reponse,i-2,2)
 If Val(reponse)<6 Or Val(reponse)= 99 Then
  Debug "Niveau de réception GSM insuffisant... Problème !",Cr
 Else
  Debug "Niveau de réception GSM (1-31): ",Dec(Val(reponse)),Cr
  Exit Do
 Endif
Loop

' ----------------- Regarde si un SMS est arrivé ? ----------------------------------------

Bclr 1,0                                           ' RAZ buffer réception
Putstr 1,"AT+CMGR=1",13,10                         ' Envoi commande "AT+CMGD" pour voir si SMS de reçu ?
Delay 2000
reponse = Getstr(1,99)                             ' Récupère réponse du GSM

If Mid(reponse,19,5) = "ERROR" Then                ' Regarde si il y a un SMS d'arrivé ?
  Debug "Aucune commande de recue",Cr
Else
  Debug "Réception d'un SMS...",Cr
  i = 20                                           ' Oui on a reçu un SMS
  Do                                               ' Recherche commande dans le SMS recu
   If Mid(reponse,i,5)="s1=on" Then                ' regarde si reçu SMS pour activer la sortie 1 ?
     sortie1=1
     Exit Do
   Endif
   If Mid(reponse,i,6)="s1=off" Then               ' regarde si reçu SMS pour désactiver la sortie 1 ?
     sortie1=0
     Exit Do
   Endif
   If Mid(reponse,i,5)="s2=on" Then                ' regarde si reçu SMS pour activer la sortie 2 ?
     sortie2=1
     Exit Do
   Endif
   If Mid(reponse,i,6)="s2=off" Then               ' regarde si reçu SMS pour désactiver la sortie 2 ?
     sortie2=0
     Exit Do
   Endif
   i = i+1                                         ' Passe au caractère suivant dans le SMS
   If i> 120 Then Exit Do                          ' Sort de la boucle si SMS reçu non reconnu
  Loop
  Out 5,sortie1                                    ' Mise à jour des sorties
  Out 6,sortie2                                    ' Mise à jour des sorties
  If i < 120 Then
    Debug "SMS de commande valide... Envoi d'un SMS de confirmation",Cr

    ' ---------------- Commence par effacer le SMS en mémoire en position N° 1  ----------------------------------------

    Putstr 1,"AT+CMGD=1",13,10                     ' Envoi commande "AT+CMGD"
    Delay 2000
    Bclr 1,0                                       ' RAZ buffer réception

    '-------------------- Envoi d'un SMS de confirmation ----------------------------------------
```

```
      Putstr 1,"AT+CMGS=06xxxxxxxx",13,10        ' Envoi commande pour SMS vers ce N° de téléphone
      Delay 1500
      Putstr 1,"Ordre bien recu !",13,10,13,10
      Putstr 1,"Etat des sorties:",13,10,13,10
      If sortie1 = 1 Then
        Putstr 1,"Sortie 1: ON",13,10
      Else
        Putstr 1,"Sortie 1: OFF",13,10
      Endif
      If sortie2 = 1 Then
       Putstr 1,"Sortie 2: ON",13,10
      Else
       Putstr 1,"Sortie 2: OFF",13,10
      Endif
      Putstr 1,&h01A                             ' Envoi du SMS
      Delay 9000
      Putstr 1,&h01B,13,10,"AT",13,10            ' Caractère ESC
                    Delay 2000
    Else
      Debug "SMS non valide...",Cr
    Endif
  Endif
Loop
```

NOTE D'APPLICATION # 42.
Robot ludique « Circular-Bot »

Cette note d'application va vous permettre de réaliser un robot mobile ludique entièrement programmable doté d'une multitude de possibilités. Ce dernier sera réalisé à partir d'un châssis « POLOLU » composé d'un disque circulaire en plastique translucide associé à un bloc double moteur avec engrenages de réduction et « ball-caster » (sorte de roue folle). Le robot appelé « Circular-Bot » sera capable dans sa version de base de se déplacer en évitant les obstacles qui se dressent devant lui grâce à 4 phototransistors et 4 leds émettrices infrarouges qui lui serviront d'yeux (2 sur le côté du robot et 2 devant lui). En plus de ces possibilités, le robot peut être agrémenté de très nombreuses options telles qu'un capteur de détection infrarouge passif capable de déceler un déplacement humain (ou animal) ou encore d'une boussole électronique permettant au robot de se repérer ou encore d'un buzzer permettant au robot de se manifester et surtout au choix d'un modem radio 433 MHz ou à technologie ZigBee™ qui vous permettra via un logiciel sur PC d'une part d'envoyer des ordres de commandes au robot à distance mais aussi de recevoir des données télémétriques de sa part en retour.

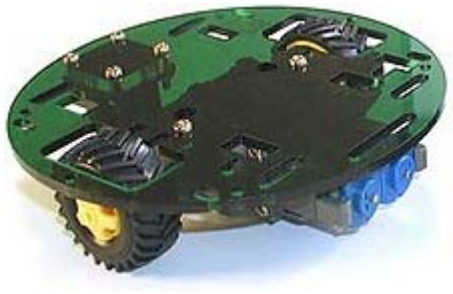
Tout est à votre disposition dans cette note d'application (schéma théorique, liste du matériel, schéma d'implantation du circuit imprimé, logiciel de commande sur PC…) afin que vous puissiez réaliser une « base ludique » éducative vous pourrez modifier à volonté pour ajouter de nouvelles possibilités de comportement au robot.

Notions abordées :
- Conversion « analogique/numérique »
- Gestion communication série (via modem radio)
- Gestion de signaux PWM (pour pilotage des moteurs)
- Gestion communication I2C™ (pour pilotage de la boussole électronique)

Matériel nécessaire :
- Un châssis de robot ludique « POLOLU »
- Module « CB220 »
- Voir la liste complète des composants nécessaires (présentée ci-après) en fonction des possibilités que vous voudrez offrir à votre robot.

Préparation matérielle :

Celle-ci nécessitera en premier lieu d'acquérir un châssis de robot mécanique « Pololu » dont vous pouvez voir ici la vue du dessus et du dessous.



Il vous faudra ensuite réaliser le circuit imprimé de la platine électronique de commande du robot. Vous trouverez ces documents dans le fichier proposant les « sources » des notes d'applications des CUBLOC (ce fichier est téléchargeable sur notre site Internet). Le schéma théorique du robot ainsi que le dessin du circuit imprimé et le programme de pilotage via les modems radio et le PC ont été réalisés via les logiciels « sPlan », « Sprint Layout » et « Profilab Expert » pour lesquels nous vous invitons à télécharger les versions de démos afin que vous puissiez visualiser les documents en détail sur l' écran de votre PC. Si vous disposez des versions « commerciales » de ces logiciels, vous pourrez alors également adapter et modifier entièrement à votre convenance les documents du robot « Circular-Bot ».

Analyse du schéma théorique :

Le schéma théorique donnée ci-après correspond à la version « complète » du robot « Circular-Bot » (c'est à dire avec toutes les options). Celui-ci est conçu sur la base d'un « CB-220 » associé à 2 circuits intégrés (LB1630) spécialisés dans la gestion de moteur « cc ». Ces circuits sont directement pilotés par 2 sorties PWM du « CB220 » (tandis que 2 autres sorties du CUBLOC permettrons de sélectionner le sens de rotation des moteurs). Le CUBLOC pilote également via un étage à transistor un buzzer (ce dernier pourra être avec ou sans oscillateur). Une boussole électronique « CMP03 » est reliée au port P8 et P9 du « CB220 » reconfiguré en bus I2C™ tandis qu'un détecteur de mouvement infrarouge passif « PI8377 » injectera son signal de sortie sur une des entrées du CUBLOC. 4 Leds infrarouges sont pilotées via un étage à transistor par une sortie du CB220 tandis que 4 étages à phototransistors « rentrent » sur 4 entrées analogiques du CUBLOC. Les 2 ports séries du « CB220 » sont respectivement attribué au connecteur de programmation et au pilotage (au choix) d'un modem radio 433 MHz ou d'un modem ZigBee™ (avec étage de régulation 3,3 V dans ce cas).

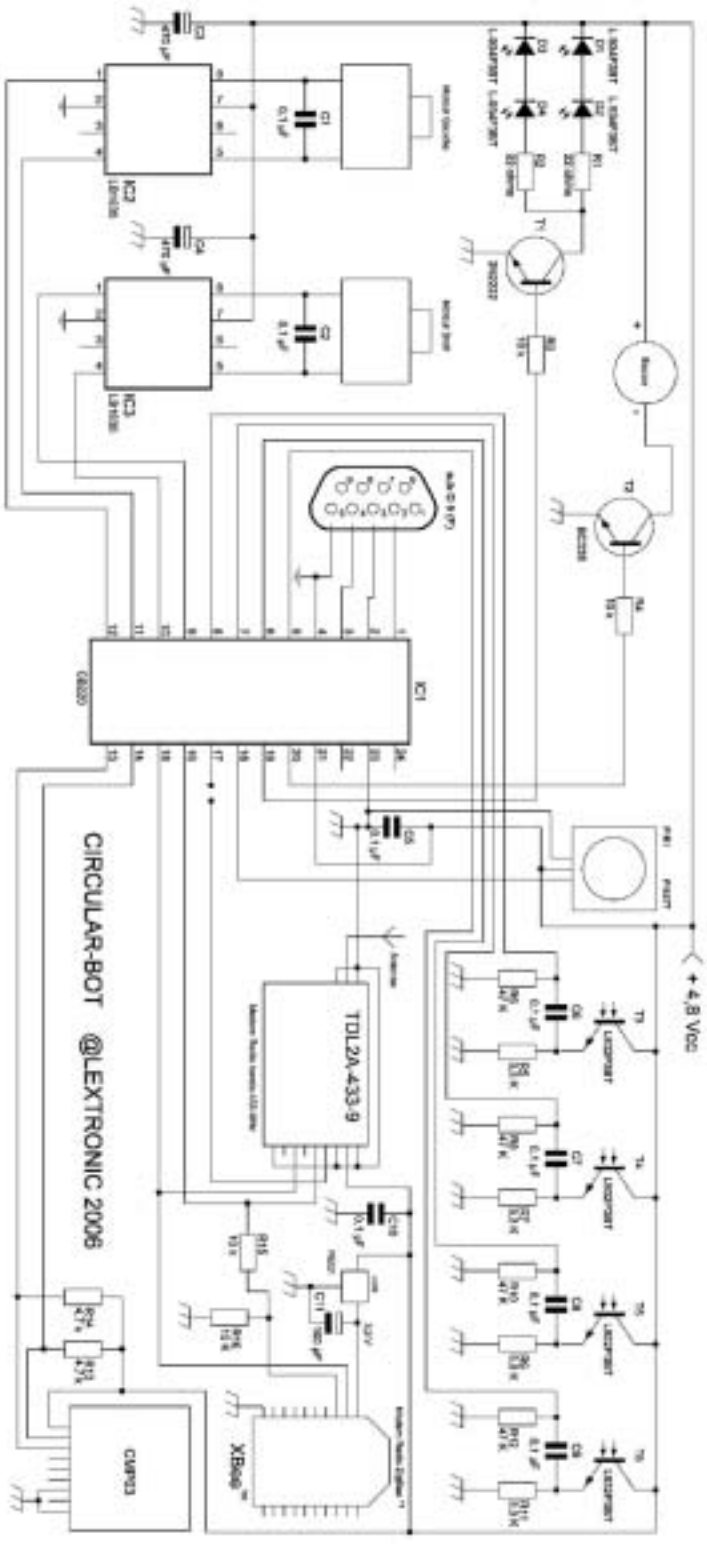
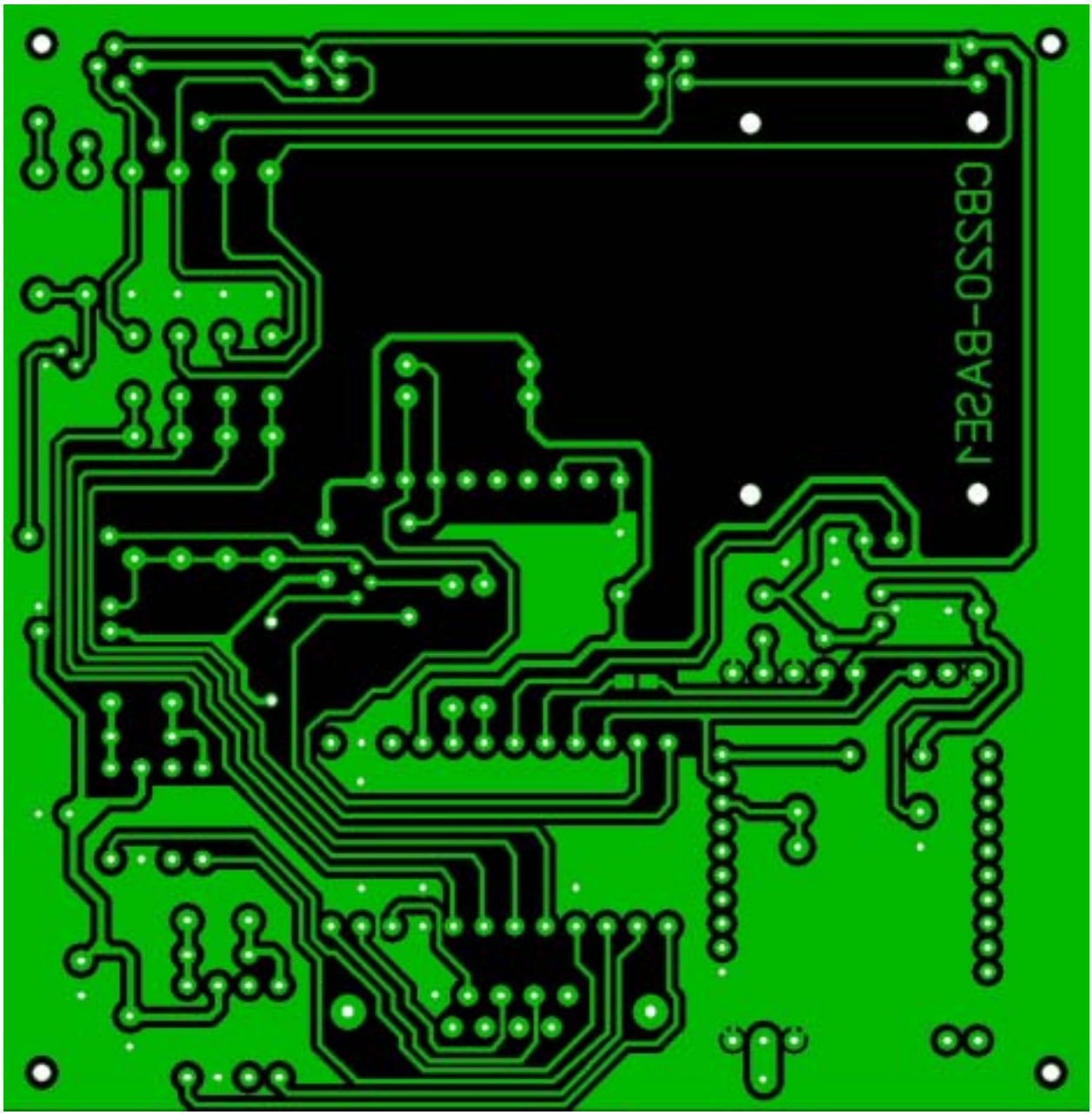


Schéma théorique complet du robot « Circular-Bot »
(Le fichier détaillé au format « sPlan » est dispo sur notre site Internet)



Tracé du circuit imprimé du robot « Circular-Bot »
(Les fichiers détaillés au format Gerber™ + « Sprint-Layout » sont dispos sur notre site Internet)

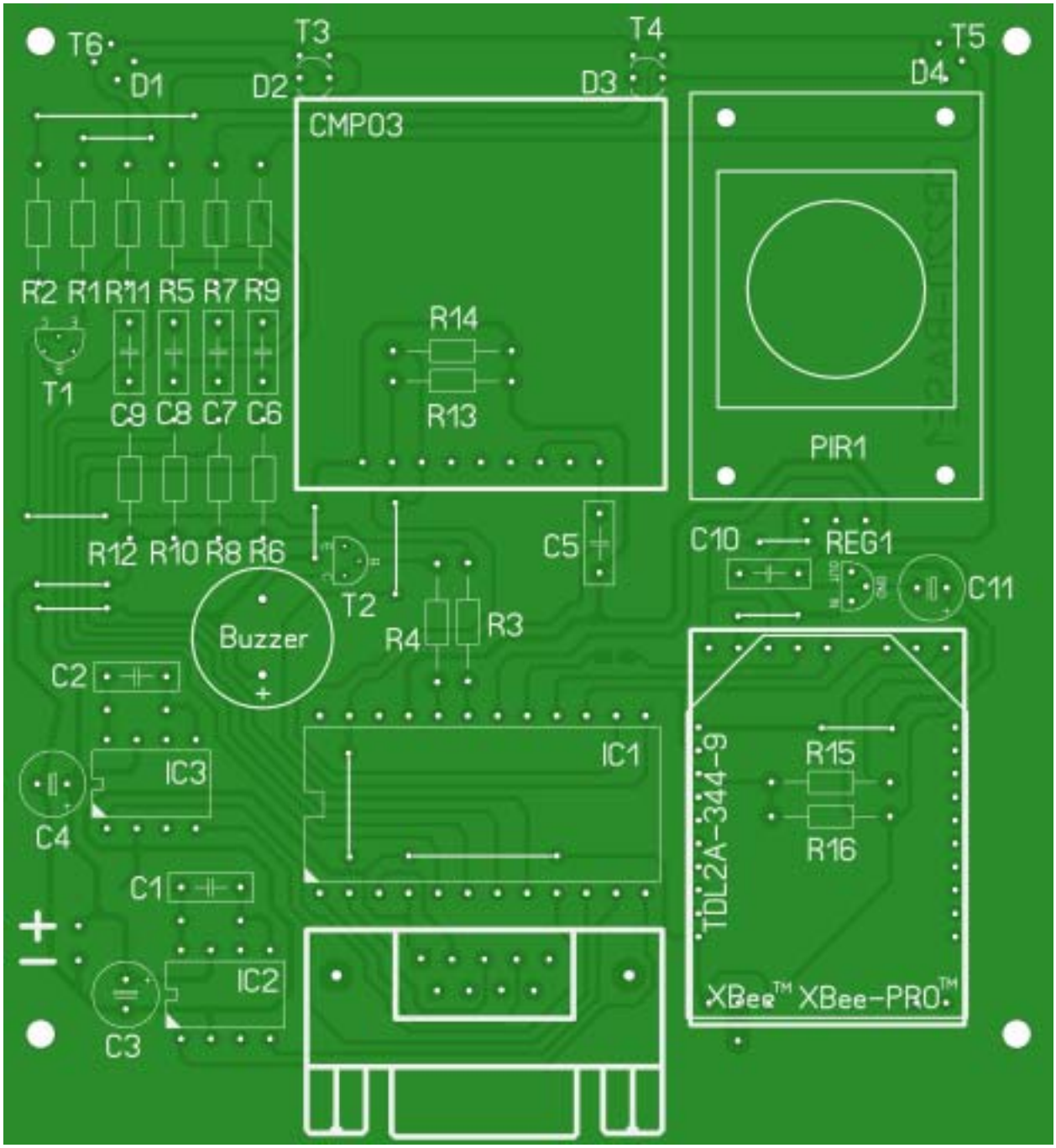


Schéma de l'implantation de tous les composants du robot « Circular-Bot »
(Les fichiers détaillés au format Gerber™ + « Sprint-Layout » sont dispos sur notre site Internet)

Liste des composants du robot « Circular-Bot »

IC1 : CB220
IC2 et IC3 : LB1630

R1 : 22 ohms
R2 : 22 ohms
R3 : 10 Kohms (si vous utilisez le buzzer)
R4 : 10 Kohms
R5 : 3,3 Kohms
R6 : 47 Kohms
R7 : 3,3 Kohms
R8 : 47 Kohms
R9 : 3,3 Kohms
R10 : 47 Kohms
R11 : 3,3 Kohms
R12 : 47 Kohms
R13 : 4,7 Kohms (si vous utilisez la boussole électronique « CMP03 »)
R14 : 4,7 Kohms (si vous utilisez la boussole électronique « CMP03 »)
R15 : 10 Kohms (si vous utilisez le modem radio « Xbee™ »)
R16 : 10 Kohms (si vous utilisez le modem radio « Xbee™ »)

C1 : 0,1 µF céramique
C2 : 0,1 µF céramique
C3 : 470 µF chimique
C4 : 470 µF chimique
C5 à C10 : 0,1 µF céramique
C11 : 100 µF chimique (si vous utilisez le modem radio « Xbee™ »)

D1 à D4 : Led IR émettrice « L-934F3BT »

T1 : 2N2222
T2 : BC338 (si vous utilisez le buzzer)
T3 à T6 : Phototransistor « L-932P3BT »

REG1 : LE33Z (régulateur 3,3 V - si vous utilisez le modem radio « Xbee™ »)

Support d'accus : BH341BS
4 x Accus 1,2 V / 2,3 A

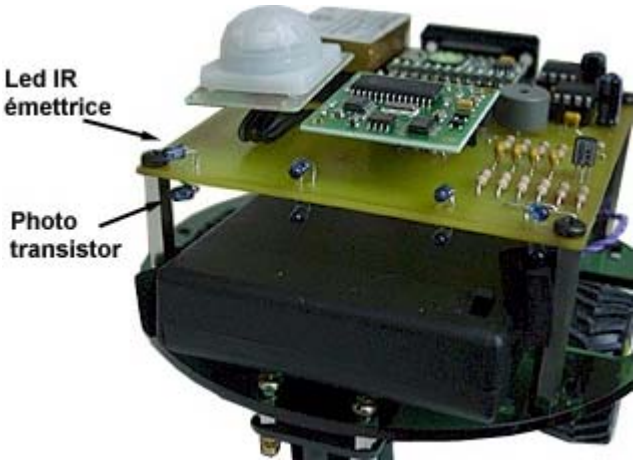
Base « POLOLU »

Options :
Buzzer (avec ou sans oscillateur incorporé)
PIR1 : Module de détection infrarouge passif PI8377
Modem radio TDL2A-433-9
Modem radio Xbee™
CMP03 : boussole électronique

Réalisation du robot « Circular-Bot » dans sa version de base

Dans sa version de base le robot « Circular-Bot » sera capable de se déplacer en évitant les obstacles. Pour réaliser ce dernier, seules les composants présentés sur la figure de droite seront nécessaires. Une fois assemblée, montez la platine sur la base « POLOLU » à l'aide de 4 entretoises. Pour alimenter le montage on utilisera un bloc support de référence « BH341BS ». Ce dernier dispose d'un interrupteur et de la possibilité de recevoir 4 accus de type AA. A ce titre il vous faudra aussi vous procurer 4 accus 1,2 V de capacité maximale (2,3 ou 2,5 A par exemple). A noter que le montage ne dispose d'aucune protection contre les inversions de polarité il vous faudra donc impérativement **vérifier la tension d'alimentation sous peine de destruction du montage**.

Prenez également soin de n'utiliser que des batteries de 1,2 V afin d'obtenir une tension totale de 4,8 V. N'utilisez JAMAIS de piles 1,5 V sans quoi vous obtiendrez une tension totale de 6 V pouvant être fatale pour l'électronique du robot. Dans un même ordre d'esprit, ne rechargez JAMAIS les batteries sur le robot afin d'éviter toute surtension. Positionnez enfin le bloc des accumulateurs à l'avant du robot pour l'équilibrer.



Les diodes d'émission infrarouges devront être montées sur le dessus de la platine à 90° et légèrement surélevées. La patte la plus longue sera connectée au (+). Les phototransistors dervont être montés côté soudure à l'opposé des diodes émettrices et également à 90° et légèrement surélevées. En absence d'obstacle, l'émission infrarouge de la diode ne parviendra pas au phototransistor associé. En présence d'un obstacle, l'émission infrarouge se reflètera dans ce dernier et ira « frapper » dans le phototransistor.

Plus l'obstacle sera près, plus la tension en sortie du phototransistor sera importante et inversement (à condition que l'obstacle soit à la portée des diodes et phototransistors et que la couleur et la nature de l'obstacle reflètent les émissions infrarouges). C'est ainsi que le robot « Circular-Bot » pourra se déplacer seul en analysant les 4 sorties des phototransistors via 4 de ses entrées de conversion analogique/numériques.

Gardez enfin à l'esprit que le robot « Circular-Bot » n'est pas un jouet et qu'il est impératif de le laissé éloigné des petits enfants et plus particulièrement des enfants inférieurs à 3 ans qui pourraient avaler ou inhaler les petites parties qui le compose.

Une fois le robot assemblé et le montage entièrement vérifié, saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « pololu2 »).

Principe de fonctionnement de l'application :
Le programme commence par la déclaration des variables et par l'initialisation des ports (en entrées ou en sortie) du « CB220 ». On initialisera également les ports du CUBLOC même si les composants optionnels ne sont pas connectés (au cas où vous les connecteriez pas la suite). Le programme configure ensuite les circuits intégrés de gestion de moteur pour une marche avant (mais avec une vitesse de rotation nulle). Aidez-vous de la note d'application N° 33 de ce même document pour bien assimiler le fonctionnement des circuits de contrôle de moteur « LB1630 ». La variable « mode » sera utiliser pour indiquer au programme comment le robot devra se comporter. Ici on l'initialise avec la valeur 0 en considérant ainsi que les moteurs doivent être à l'arrêt. Après une temporisation de 5 secondes, on émet un bip sonore dans le buzzer pour signaler que le robot est prêt à fonctionner (ceci vous laisse le temps de mettre l'alimentation sur le montage et de vous éloigner des capteurs). Si vous utilisez un buzzer avec oscillateur intégré (ce qui n'est pas notre cas), il vous faudra simplement utiliser l'instruction OUT (tantôt à 1 et tantôt à 0 avec une temporisation entre les 2 pour générer le bip sonore). A dire vrai, la sortie buzzer n'est pas vraiment exploitée dans notre réalisation et vous pourrez vous-même définir d'autres situations à partir desquels le buzzer devra être à nouveau sollicité (en cas de détection d'obstacle par exemple).

A partir de ce stade, le programme entre dans une boucle sans fin dans laquelle il va commencer par alimenter les 4 diodes émettrices infrarouges, puis récupérer les valeurs analogiques des 4 tensions présentes en sortie des montages à phototransistors pour ensuite désactiver l'alimentation des diodes infrarouges (afin d'éviter une consommation excessive de ces dernières – ne les laissez pas alimenter en permanence). En cas de valeur analogique inférieur à 20, le programme initialise les valeurs lues à « 0 » afin de filtrer les réceptions « parasites » des phototransistors.

La suite du programme vérifie s'il n'y a pas de détection d'obstacle sur les 2 phototransistors latéraux et va ensuite en fonction de l'état de la variable « mode » donner des consignes aux moteurs. Ainsi à la mise sous tension, si vous avez pris soin d'éloigner le robot de tout obstacle et de ne pas laisser vos mains devant les capteurs le robot restera immobile. Pour lui donner « vie » il vous suffira alors d'approcher un obstacle ou vos mains devant les capteurs latéraux. Dès lors, la variable « mode » passera à la valeur « 2 » ou « 3 » suivant que l'obstacle est plus près du capteur latéral « droit » ou « gauche ».

En mode « 2 », le robot a détecté un obstacle à gauche et dans ce cas, il bloquera la roue droite et activera la roue gauche afin de tourner à droite pour contourner l'obstacle.

En mode « 3 », le robot a détecté un obstacle à droite et dans ce cas, il bloquera la roue gauche et activera la roue droite afin de tourner à gauche pour contourner l'obstacle.

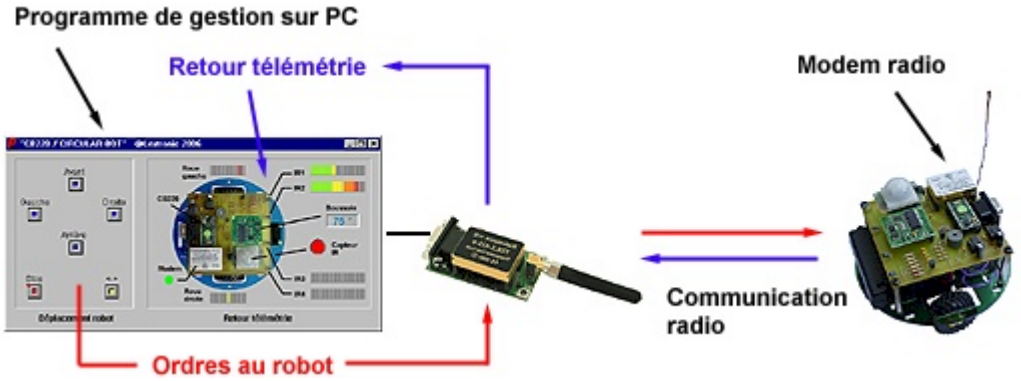
Un autre test donne l'ordre au robot d'avancer tout droit (mode = « 1 ») dès que plus aucun obstacle ne sera détecté par les 4 capteurs (ceci permettra au robot de « sortir » de son détour dès que l'obstacle aura été contourné). Bien d'autres « stratégies » peuvent être mises au point sur le même principe. Nous vous laissons le soin de les développer (c'est là que réside tout l'intérêt de cette application).

Dernière précision : vérifiez que le bloc moteur du robot soit parfaitement assemblé et ne génère aucun accrochage sans quoi l'appel de courant lors du démarrage des moteurs sera tel que le microcontrôleur sera réinitialisé et que les moteurs n'avanceront pas (ceci peut aussi être le cas si vous utilisez d'autres types de moteur). La solution de « secours » sera alors d'utiliser 2 sources d'alimentation distincte une pour les moteurs et une pour l'électronique.

```
'###################################
'#        Robot « Circular-Bot »                #
'#           « Version simple »                 #
'#     @Lextronic 2006 - 12/11/2006        #
'###################################

Const Device = CB220

Dim mode As Byte                                    ' Mode de fonctionnement
Dim ir1 As Integer                                  ' Variable distance obstacle IR1
Dim ir2 As Integer                                  ' Variable distance obstacle IR4
Dim ir3 As Integer                                  ' Variable distance obstacle IR3
Dim ir4 As Integer                                  ' Variable distance obstacle IR2

Input 0                                             ' Port 0 en entree <- Capteur IR1
Input 1                                             ' Port 1 en entree <- Capteur IR4
Input 2                                             ' Port 2 en entree <- Capteur IR3
Input 3                                             ' Port 3 en entree <- Capteur IR2
Low 4                                               ' Port P4 en sortie (sens de rotation moteur droit)
Low 5                                               ' Port PWM 0 en sortie (avec niveau bas)
Low 6                                               ' Port PWM 1 en sortie (avec niveau bas)
Low 7                                               ' Port P7 en sortie (sens de rotation moteur gauche)
Input 13                                            ' Port 3 en entree <- Capteur presence
Low 14                                              ' Led IR emettrices OFF
Low 15                                              ' initialisation port du buzzer

mode = 0                                            ' Mode Robot a l'arret
Out 7,0                                             ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
Out 4,0                                             ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
Pwm 0,0,1024                                        ' Stop moteur droit
Pwm 1,0,1024                                        ' Stop moteur gauche
Delay 5000
Beep 15,300                                         ' Signal que le robot est pret par un bip sonore

Do                                                  ' Boucle principale avec recuperation distance des obstacles

   High   14                                        ' Led IR emettrice ON
   ir1 = Adin(0)                                    ' Recupere distance obstacle capteur IR1
   ir2 = Adin(3)                                    ' Recupere distance obstacle capteur IR2
   ir3 = Adin(2)                                    ' Recupere distance obstacle capteur IR3
   ir4 = Adin(1)                                    ' Recupere distance obstacle capteur IR4
   Low 14                                           ' Led IR emettrices OFF

   If ir1 < 20 Then ir1 = 0                         ' Filtrage detection IR parasite
   If ir2 < 20 Then ir2 = 0                         ' Filtrage detection IR parasite
   If ir3 < 20 Then ir3 = 0                         ' Filtrage detection IR parasite
   If ir4 < 20 Then ir4 = 0                         ' Filtrage detection IR parasite

   '---------------------------- Gestion des obstacles ---------------------------------

    If ir1 Or ir4 > 0 Then                          ' il y a un obstacle de detecte
      mode = 3                                      ' Force le robot a tourner a gauche jusqu'a plus d'obstacle
      If ir1 > ir4 Then mode = 2                    ' Force le robot a tourner a droite jusqu'a plus d'obstacle
    End If

    If mode <> 0 Then                               ' Test si on est pas a l'arret
      If ir1+ir2+ir3+ir4 = 0 Then mode = 1          ' Avance tout droit si plus d'obstacle
    End If

     '---------------------------- Gestion des moteurs ---------------------------------

    Select Case mode                                ' Regarde dans quel mode on est ?

       Case 2                                       ' Force robot a tourner a droite pour eviter obstacle
```

```
            Out 7,0                         ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
            Out 4,0                         ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
            Pwm 0,0,1024                    ' Moteur droit stop
            Pwm 1,450,1024                  ' Vitesse de rotation moteur gauche

          Case 3                            ' Force robot a tourner a gauche pour eviter obstacle
            Out 7,0                         ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
            Out 4,0                         ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
            Pwm 0,350,1024                  ' Vitesse de rotation moteur droit
            Pwm 1,0,1024                    ' Moteur gauche stop

          Case 1                            ' Cas Robot marche avant
            Out 7,0                         ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
            Out 4,0                         ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
            Pwm 0,350,1024                  ' Vitesse de rotation moteur droit
            Pwm 1,450,1024                  ' Vitesse de rotation moteur gauche

        End Select
Loop
```

Réalisation du robot « Circular-Bot » dans sa version « complète »
Dans cette version on utilisera des modems radio permettant de réaliser une liaison bidirectionnelle entre un PC associé à un logiciel « spécialisé » et la base robotique afin de pouvoir à la fois envoyer des ordres au robot (pour choisir comment il va se déplacer), mais également pour recevoir en retour les informations des capteurs et des moteurs afin de prendre connaissance (même sans le voir) du comportement et de l'environnement du robot.



Le circuit imprimé de la base robotique est prévue pour recevoir au choix un modem Radiometrix™ de type « TDL2A-433-9 » ou un modem de type Maxstream™ « Xbee™ » à technologie ZigBee™.

Si vous utilisez le modem « TDL2A-433-9 », il vous faudra alors connecter un module « TDL2I-433 » sur le port série du PC afin que ce dernier puisse communiquer avec le modem embarqué du robot. Consultez les notes d'application N° 11 et 12 pour plus d'infos.

Si vous utilisez le modem « Xbee™ » il vous faudra alors connecter un module « Xbee™ en boîtier » sur le port série du PC afin que ce dernier puisse communiquer avec le modem embarqué du robot. De plus la platine électronique du robot devra recevoir un régulateur supplémentaire afin de pouvoir générer les 3,3 V nécessaire au fonctionnement du modem « Xbee™ ». Consultez la note d'application N° 35 pour plus d'infos.

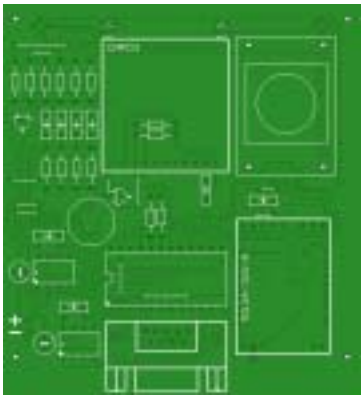
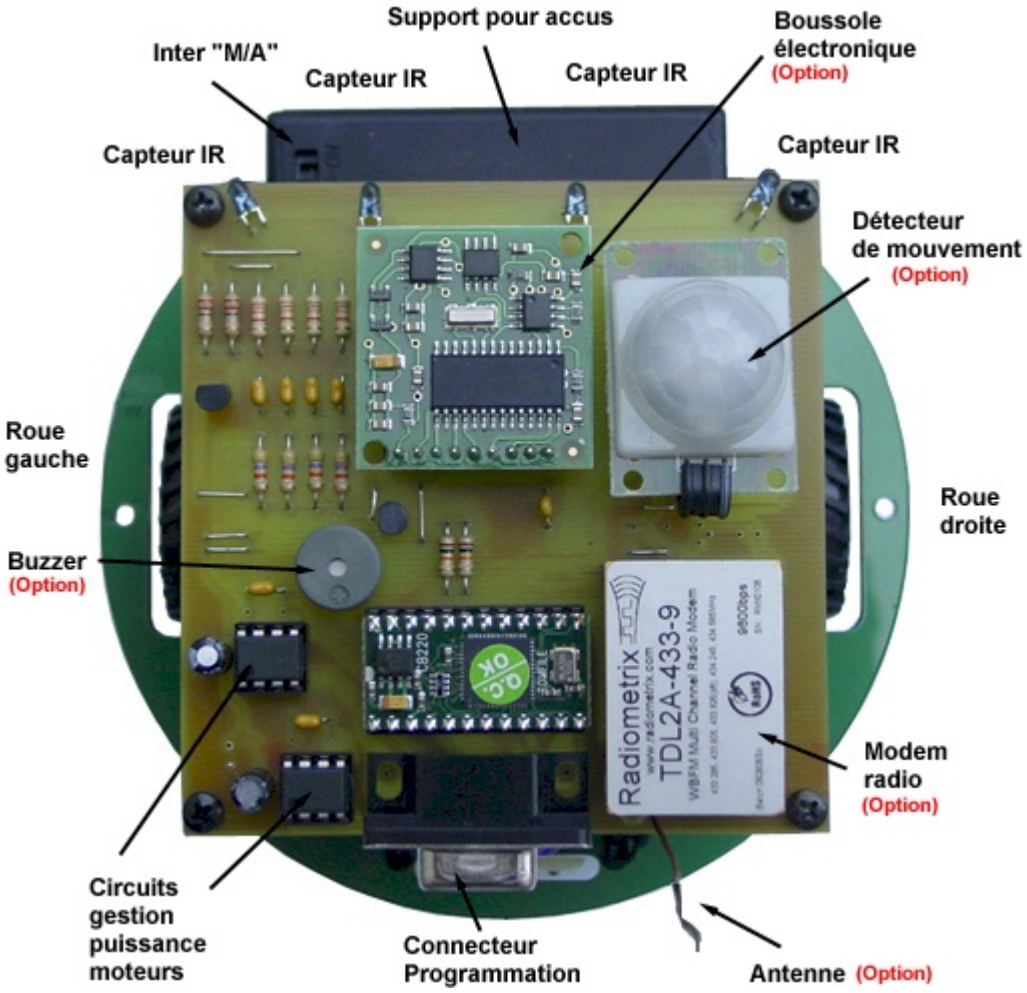


Schéma de montage pour modem « TDL2A-433-9 »

Schéma de montage pour modem « Xbee™ »

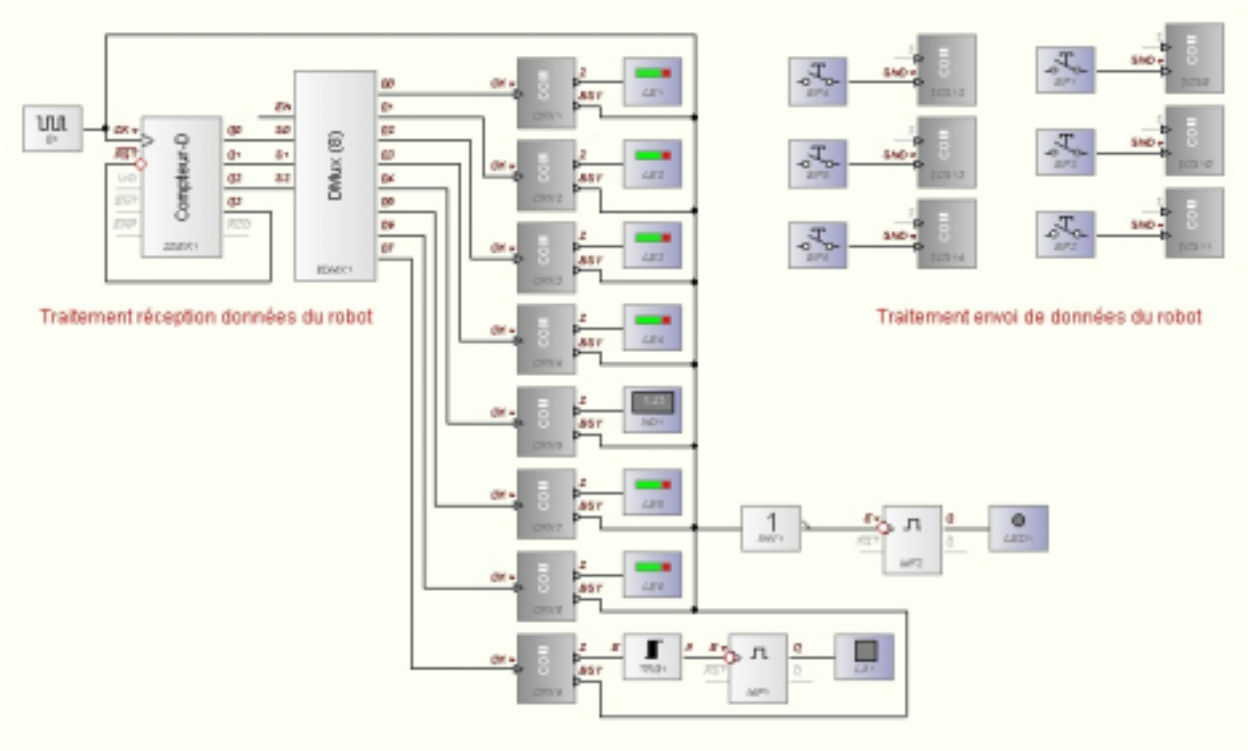
Le raccordement des modems au CUBLOC est très simple et se limite aux connexions des broches TX et RX du port série N° 1. A noter qu'une des broches du CUBLOC est prévue si nécessaire pour être relié via un pont de soudure à la broche de configuration du modem « TDL2A-433-9 » afin que vous puissiez si nécessaire demander au « CB220 » de reconfigurer le modem radio afin de modifier sa fréquence et son code « site » (idéal si vous voulez réaliser un « flotte » de plusieurs robot « Circular-Bot » devant dialoguer avec une base ou les uns avec les autres !



La photo ci-dessus montre la platine équipée de toutes les options avec le modem Radiometrix™ « TDL2A-433-9 ». Certains trous sont libres pour pouvoir recevoir un étage de régulation 3,3 V au cas où vous voudriez monter un modem ZigBee™ à la place du modem Radiometrix™.

Une fois le robot assemblé et le montage entièrement vérifié, saisissez ensuite le programme présenté ci-après (ce dernier est disponible sur notre site Internet : www.lextronic.fr ou sur notre CD-ROM sous le nom « pololu3 »).

Principe de fonctionnement de l'application (côté « PC ») :
Le programme de pilotage du robot du PC a été développé en quelques minutes sans aucune ligne de programme grâce au logiciel « Profilab-Expert ». Ce logiciel est un générateur d'application graphique qui permet de véritables « petites merveilles » pour un prix très compétitif. Une fois l'application réalisée, vous pourrez la compiler et disposer d'un exécutable indépendant librement diffusable (vous retrouverez le programme « graphique » et l'exécutable compilé dans le fichier proposant les « sources » des notes d'applications des CUBLOC (ce fichier est téléchargeable sur notre site Internet). Si vous disposez de la version « commerciale » de « Profilab-Expert », vous pourrez ainsi modifier le programme de gestion du robot (côté PC) à volonté.

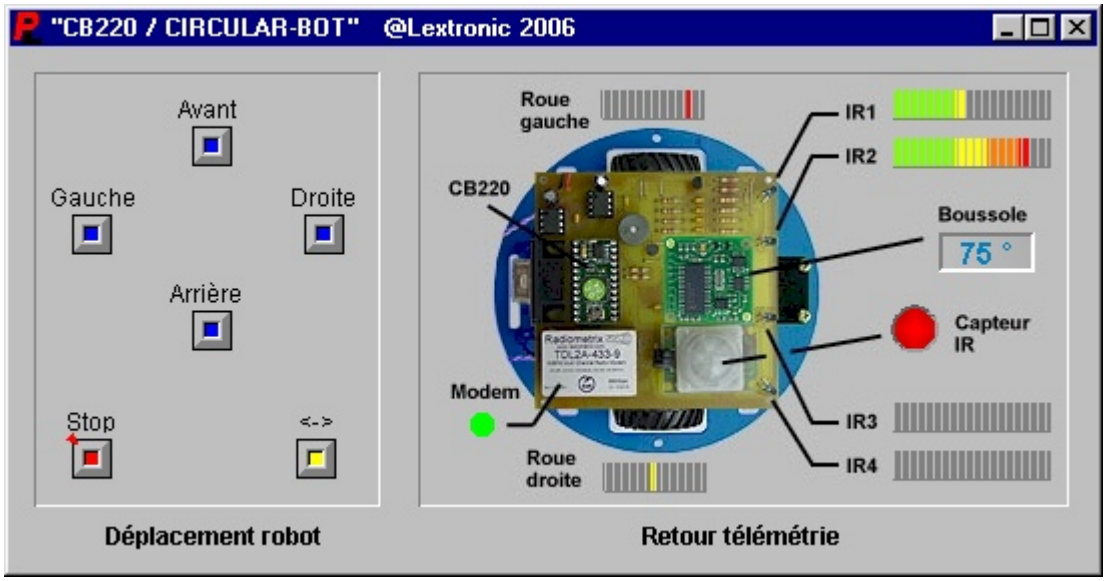


Voici à quoi ressemble les blocs graphiques utilisés pour développer l'application côté « PC ». L'application peut être divisée en 2 parties :

La première (permettant l'envoi d'ordres de commande au PC) consiste suite à l'action sur des bouton-poussoirs via la souris d'envoyer des octets sur le port série du « PC » (ces ordres seront envoyés au robot via le modem connecté au port série du PC).

La seconde partie consiste à venir lire le contenu du buffer de réception du port série du PC (lequel reçoit des informations du robot via le modem radio) et en fonction des valeurs reçues d'afficher ces données sur différents bargraphs et voyant.

N'hésitez pas à télécharger la démo de « Profilab-Expert » afin de mieux assimiler le fonctionnement. La documentation de ce dernier est toute en Français !



Voici ensuite à quoi ressemble l'écran de contrôle du logiciel ainsi réalisé avec « Profilab-Expert ». A gauche se trouve le panneau de commande du robot qui vous permettra via le modem relié au port série du PC d'envoyer des ordres pour lui demander d'avancer tout droit, d'effectuer un virage à droite ou à gauche, de reculer, de s'arrêter ou de faire demi-tours sur lui-même (avec le bouton jaune).

La partie de droite est réservée au retour d'information télémétrique du robot. Le dessin du robot vu de dessus a été ajouté sur le fond de l'écran afin de bien savoir à quoi correspondent les différents voyants et bargraphs.

Ainsi en bas à gauche, vous disposerez d'un voyant « vert » qui clignotera lorsque le modem du « PC » recevra des données de la part du modem du robot (en cas contraire, le voyant reste éteint).

De part et d'autre du robot, 2 bargraphs donne l'état des roues du robot. Vous pourrez ainsi savoir (même sans le voir) si le robot avance, recul, est à l'arrêt, effectue un virage à droite ou à gauche ou fait un demi-tour sur lui-même.

Au centre à gauche se trouve un voyant rouge qui vous indiquera lorsqu'il est allumé que le robot a détecté un mouvement humain ou animal à proximité.

Un afficheur vous donne aussi le cap retourné par la boussole embarquée sur le robot.

Vous disposez enfin de 4 bargraphs représentant l'état individuel de chaque phototransistor afin de prendre connaissance de l'environnement du robot et de la présence d'obstacle.

Ainsi sur la copie d'écran ci-dessus, on peu voir que le PC reçoit bien des données radio de la part du robot et que ce dernier a détecté un déplacement humain à proximité, qu'il est au cap 75° et qu'il a détecté un obstacle sur sa gauche (l'obstacle est plus proche du détecteur IR2 que du détecteur IR1). On voit également très bien que la roue gauche est activée en marche avant et que la roue droite est stoppée (le robot est en fait en train d'effectuer un virage à droite pour éviter l'obstacle !).

Sur notre application, la boussole et le capteur de mouvement IR sont juste exploités afin de connaître leur état depuis le PC, mais vous pourrez aisément modifier le programme du robot pour par exemple activer le fonctionnement du robot uniquement en cas de mouvement « humain » dans la pièce ou bien d'autres choses encore…

Principe de fonctionnement de l'application (côté « Robot ») :
Le programme utilise le même mode de fonctionnement que le robot « standard » en ce qui concerne les capteurs IR (pour la détection des obstacles) et la gestion des moteurs. Toutefois le programme récupère désormais en plus les données du capteur IR de mouvement humain et de la boussole via le bus I2C™ (vous disposez d'une description complète de la boussole dans la note d'application N° 6 des CUBLOC). Toutes les données ainsi collectées seront « converties » sous la forme de chaînes ASCII. C'est à dire que si la variable numérique récupérant l'angle de la boussole est de 75, on utilisera une fonction spécialement développée pour récupérer 75 dans une chaîne alphanumérique à laquelle on ajoutera à la fin de cette chaîne alphanumérique par une lettre « A », « B »……. Ou « G ». La raison de cette conversion réside dans le fait que le programme développé sur le PC à l'aide de « Profilab-Expert » va lire le buffer de réception du PC et récupérer les caractères ASCII se trouvant avant les lettres « A », « B »……. Ou « G ». En fait, ces lettres servent de repères pour que le logiciel puisse savoir à quel bagraph il doit attribuer les valeurs reçues (sous la forme de chaîne ASCII).

Ainsi les on ajoutera le caractère « A » après les données relatives au capteur IR1, le caractère « B » après les données relatives au capteur IR2, le caractère « C » après les données relatives au capteur IR3, le caractère « D » après les données relatives au capteur IR4, le caractère « E » après les données relatives à la boussole, le caractère « F » après les données relatives au sens de rotation du moteur droit, le caractère « G » après les données relatives au sens de rotation du moteur gauche et enfin le caractère « H » après les données relatives à la détection du capteur de mouvement humain IR.

Ceci permettra de créer une longue chaîne de caractères alphanumériques par exemple du style : 10A25B0C0D75E511F511F1G

Cette chaîne sera envoyée par le robot au modem radio via le port série afin que le programme du PC puisse en la recevant « décoder » celle-ci afin d'afficher les valeurs correspondantes sur les bons voyants et bargraphs.

A noter que lorsque le programme du robot pilote les moteurs, il va attribuer une chaîne alphanumérique représentant le reflet de l'état des moteurs droit et gauche. Ainsi lorsque le moteur gauche est à l'arrêt, le programme attribuera la chaîne « 511 » avant la lettre « G » afin que le programme du PC affiche une barre à mi-course du bargraph (dont l'échelle est comprise entre 0 et 1023) afin que l'on indique que le moteur est à l'arrêt. Si le moteur est activé en marche avant alors le programme attribuera la chaîne « 900 » avant la lettre « G » afin que le programme du PC affiche une barre vers la fin du bargraph et que l'on puisse voir que le moteur est en marche avant. Si le moteur est activé en arrière alors le programme attribuera la chaîne « 200 » avant la lettre « G » afin que le programme du PC affiche une barre vers le début du bargraph et que l'on puisse voir que le moteur est en marche arrière.

A noter notre programme n'utilise qu'une seule vitesse de rotation pour les roues du robot. Toutefois selon le même principe il serait totalement possible de moduler la vitesse du robot et de retourner sur le PC n'on plus seulement l'état des roues (arrêt / marche avant / marche arrière) mais pourquoi pas également une « représentation » de leur vitesse.

Une autre partie du programme récupère également l'état du buffer de réception du port série afin de vérifier si le robot a reçu des ordres de pilotage de la part du logiciel du PC. Ces ordres se présentent simplement sous la forme de valeurs numériques pouvant prendre la forme d'un octet de valeur 1 à 6. Pour chacune de ces valeurs, le robot effectuera une action.

Pour la valeur « 1 », le robot avancera tout droit.

Pour la valeur « 2 » le robot effectuera un virage à gauche (ce dernier va en fait stopper sa roue gauche et activer sa roue droite pendant un faible instant pour tourner, puis les 2 roues se mettrons à tourner en avant à nouveau pour que le robot avance tout droit).

Pour la valeur « 3 » le robot effectuera un virage à droite (ce dernier va en fait stopper sa roue droite et activer sa roue gauche pendant un faible instant pour tourner, puis les 2 roues se mettrons à tourner en avant à nouveau pour que le robot avance tout droit).

Pour la valeur « 4 », le robot stoppera puis effectuera une marche arrière.

Pour la valeur « 5 », le robot stoppera complètement.

Pour la valeur « 6 », le robot effectuera un demi-tour complet sur lui-même en faisant tourner pendant un bref instant une roue dans un sens et l'autre roue dans l'autre sens.

A noter que les commandes envoyées par le programme du PC peuvent nécessiter que vous sollicitiez plusieurs fois les touches de commandes du programme du PC car nous avons privilégié l'envoie rapide du retour télémétrique des données sur robot vers le PC afin d'obtenir un affichage en quasi-temps réel. Le revers de la médaille est que le modem radio est souvent utilisé en émission (de robot vers le PC). Dès lors lorsque vous envoyez un ordre du PC vers le robot ce dernier pourra déjà être en réception et ne pas réagir immédiatement.

```
'###################################
'#        Robot « Circular-Bot »                #
'#           « Version complète »               #
'#     @Lextronic 2006 - 12/11/2006        #
'###################################

Const Device = CB220

Dim i As Byte                                   ' Variable usage general
Dim mode As Byte                                ' Mode de fonctionnement
Dim ir1 As Integer                              ' Variable distance obstacle IR1
Dim ir2 As Integer                              ' Variable distance obstacle IR4
Dim ir3 As Integer                              ' Variable distance obstacle IR3
Dim ir4 As Integer                              ' Variable distance obstacle IR2
Dim st1 As String * 10                          ‘ Variable traitement de chaine
Dim st2 As String * 10                          ' Variable traitement de chaine
Dim st3 As String * 10                          ' Variable traitement de chaine
Dim st4 As String * 10                          ' Variable traitement de chaine
Dim st5 As String * 5                           ' Variable traitement de chaine
Dim st7 As String * 2                           ‘ Variable detection mouvement humain
Dim st8 As String * 5                           ' Vitesse rotation moteur droit
Dim st9 As String * 5                           ' Vitesse rotation moteur gauche
Dim st10 As String * 5                          ' Variable boussole pour telemetrie
Dim compass As Integer                          ' Variable boussole (acquisition)
Dim angle As Single                             ' Variable calcul angle boussole
Dim errorcom As Byte                            ' Gestion erreur lecture composant I2C
```

```
Opencom 1,9600,3,10,50
Set I2c 8,9                                     ' Configure Bus I2C

Input 0                                         ' Port 0 en entree <- Capteur IR1
Input 1                                         ' Port 1 en entree <- Capteur IR4
Input 2                                         ' Port 2 en entree <- Capteur IR3
Input 3                                         ' Port 3 en entree <- Capteur IR2
Low 4                                           ' Port P4 en sortie (sens de rotation moteur droit)
Low 5                                           ' Port PWM 0 en sortie (avec niveau bas)
Low 6                                           ' Port PWM 1 en sortie (avec niveau bas)
Low 7                                           ' Port P7 en sortie (sens de rotation moteur gauche)
Input 13                                        ' Port 3 en entree <- Capteur presence
Low 14                                          ' Led IR emettrices OFF
Low 15                                           ' initialisation port du buzzer

mode = 0                                        ' Robot a l'arret
Bclr 1,0                                        ' RAZ buffer rception
Delay 5000
Beep 15,300                                     ' Signal que le robot est pret par un bip sonore

Do
        '---------- Recuperation distance des obstacles ----------

  High   14                                     ' Led IR emettrice ON
  ir1 = Adin(0)                                 ' Recupere distance obstacle capteur IR1
  ir2 = Adin(3)                                 ' Recupere distance obstacle capteur IR2
  ir3 = Adin(2)                                 ' Recupere distance obstacle capteur IR3
  ir4 = Adin(1)                                 ' Recupere distance obstacle capteur IR4
  Low 14                                        ' Led IR emettrices OFF

  If ir1 < 20 Then ir1 = 0                      ' Filtrage detection IR parasite
  If ir2 < 20 Then ir2 = 0                      ' Filtrage detection IR parasite
  If ir3 < 20 Then ir3 = 0                      ' Filtrage detection IR parasite
  If ir4 < 20 Then ir4 = 0                      ' Filtrage detection IR parasite

  st1 = convert (Float ir1)+"A"                 ' Recupere valeur entiere dans chaine st1
  st2 = convert (Float ir2)+"B"                 ' Recupere valeur entiere dans chaine st2
  st3 = convert (Float ir3)+"C"                 ' Recupere valeur entiere dans chaine st3
  st4 = convert (Float ir4)+"D"                 ' Recupere valeur entiere dans chaine st4

  i = In(13)                                    ' Recupere detection mouvement humain ?
  st7 = convert (Float i)+"H"                    ' Recupere valeur entiere dans chaine st7


 '---------- Recuperation orientation boussole ----------

  I2cstart                                      ' Condition Start I2C
  errorcom = I2cwrite (&HC0)                    ' Adresse du module CMP03
  errorcom = I2cwrite (2)                       ' Selectionne l'adresse de l'angle a lire
  I2cstart                                      ' Condition Start I2C
  errorcom = I2cwrite (&HC1)                    ' Selectionne condition de lecture I2C
  compass.byte1=I2cread(0)                      ' Recupere octet poids fort de la distance
  compass.byte0=I2cread(0)                       ' Recupere octet poids faible de la distance
  errorcom = I2cwrite (&HC1)                    ' Selectionne condition de lecture I2C
  I2cstop                                       ' Condition Stop I2C
  angle= compass/10                             ' Division par 10 pour affichage 0 - 359,9
  st10 = convert (Float angle)+"E"               ' Recupere valeur entiere dans chaine st10

 '---------- Recuperation des ordres radio du PC ----------

 If Blen(1,0) <> 0 Then                         ' Regarde si on a recu un ordre radio du PC ?
  i = Get(1,1)                                   ' recupere l'ordre radio du PC ?
            Select Case i                        ' Regarde quelle est l'action  faire ?
```

```
   Case 5                                  ' Cas Robot a l'arret
      mode = 0
   Case 1                                  ' Cas Robot marche avant
      mode = 1
    Case 3                                 ' Cas Robot tourne a droite
      Out 7,0                               ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
      Out 4,0                               ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
       Pwm 0,0,1024                         ' Moteur droit stop
      Pwm 1,450,1024                        ' Vitesse de rotation moteur gauche
      st8 = "511F"                          ' Initialise valeur retour telemetrie radio
      st9 = "900G"                          ' Initialise valeur retour telemetrie radio
      Putstr 1,st1,st2,st3,st4,st10,st8,st9,st7  ' envoi infos telemetrie
      Delay 1200
      mode = 1                              ' force robot en marche avant
    Case 2                                  ' Cas Robot tourne a gauche
      Out 7,0                               ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
      Out 4,0                               ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
      Pwm 0,350,1024                        ' Vitesse de rotation moteur droit
      Pwm 1,0,1024                          ' Moteur gauche stop
      st8 = "900F"                          ' Initialise valeur retour telemetrie radio
      st9 = "511G"                          ' Initialise valeur retour telemetrie radio
      Putstr 1,st1,st2,st3,st4,st10,st8,st9,st7  ' envoi infos telemetrie
      Delay 1200
       mode = 1                             ' force robot en marche avant
    Case 4                                  ' Cas Robot marche arriere
       Gosub stop                           ' Robot Stop
       mode = 0                             ' Robot marche arrieret
   Case 6                                   ' Cas Robot demi tour
       Gosub stop                           ' Robot Stop
       Out 7,0                              ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
       Out 4,1                              ' Entree IN1 du LB1630 No 2 = 0 -> Marche arriere
       Pwm 0,350,1024                       ' Vitesse de rotation moteur droit
       Pwm 1,574,1024                       ' Vitesse de rotation moteur gauche
      st8 = "900F"                          ' Initialise valeur retour telemetrie radio
      st9 = "200G"                          ' Initialise valeur retour telemetrie radio
      Putstr 1,st1,st2,st3,st4,st10,st8,st9,st7  ' envoi infos telemetrie
      Delay 1300
     Gosub stop                             ' Robot Stop
   End Select
 End If

'---------------------------- Gestion des obstacles ---------------------------------

If ir1 Or ir4 > 0 Then                      ' il y a un obstacle de detecte
   mode = 3                                 ' Force le robot a tourner a gauche jusqu'a plus d'obstacle
  If ir1 > ir4 Then mode = 2                ' Force le robot a tourner a droite jusqu'a plus d'obstacle
End If

If mode <> 0 And mode <> 4 Then              ' Test si on est pas a l'arret ou en marche arriere ?
   If ir1+ir2+ir3+ir4 = 0 Then mode = 1     ' Avance tout droit si plus d'obstacle
End If

'---------------------------- Gestion des moteurs ---------------------------------

 Select Case mode                           ' Regarde dans quel mode on est ?
  Case 0                                    ' Cas Robot a l'arret
    Gosub stop
  Case 2                                    ' Force robot a tourner a droite pour eviter obstacle
     Out 7,0                                ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
      Out 4,0                               ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
      Pwm 0,0,1024                          ' Moteur droit stop
       Pwm 1,450,1024                       ' Vitesse de rotation moteur gauche
       st8 = "511F"                         ' Initialise valeur retour telemetrie radio
       st9 = "900G"                         ' Initialise valeur retour telemetrie radio
```

```
        Case 3                                  ' Force robot a tourner a gauche pour eviter obstacle
          Out 7,0                               ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
          Out 4,0                               ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
          Pwm 0,350,1024                        ' Vitesse de rotation moteur droit
          Pwm 1,0,1024                          ' Moteur gauche stop
          st8 = "900F"                          ' Initialise valeur retour telemetrie radio
          st9 = "511G"                          ' Initialise valeur retour telemetrie radio
        Case 1                                  ' Cas Robot marche avant
          Out 7,0                               ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
          Out 4,0                               ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
          Pwm 0,350,1024                        ' Vitesse de rotation moteur droit
          Pwm 1,450,1024                        ' Vitesse de rotation moteur gauche
          st8 = "900F"                          ' Initialise valeur retour telemetrie radio
          st9 = "900G"                          ' Initialise valeur retour telemetrie radio
        Case 4                                  ' Cas Robot marche arriere
          Out 7,1                               ' Entree IN1 du LB1630 No 1 = 1 -> Marche avant
          Out 4,1                               ' Entree IN1 du LB1630 No 2 = 1 -> Marche avant
          Pwm 0,674,1024                        ' Vitesse de rotation moteur droit
          Pwm 1,574,1024                        ' Vitesse de rotation moteur gauche
          st8 = "200F"                          ' Initialise valeur retour telemetrie radio
          st9 = "200G"                          ' Initialise valeur retour telemetrie radio
      End Select


      '---------------------- Envoi des donnees via modem radio -----------------------

       Putstr 1,st1,st2,st3,st4,st10,st8,st9,st7
       Delay 300
      Loop

      '---------------------- routine stop moteurs -----------------------

stop:
      Out 7,0                                   ' Entree IN1 du LB1630 No 1 = 0 -> Marche avant
      Out 4,0                                   ' Entree IN1 du LB1630 No 2 = 0 -> Marche avant
      Pwm 0,0,1024                              ' Stop moteur droit
      Pwm 1,0,1024                              ' Stop moteur gauche
      st8 = "511F"                              ' Initialise valeur retour telemetrie radio
      st9 = "511G"                              ' Initialise valeur retour telemetrie radio
      Putstr 1,st1,st2,st3,st4,st10,st8,st9,st7 ' envoi infos telemetrie
      Pause 500                                 ' Stop les moteurs avant marche arriere
      mode = 0
     Return
   End


      '---------------------- Fonction de conversion ---------------------------------

      Function convert (st6 As String * 10) As String * 10
       i = 1
       st5 = ""
       Do
         If Mid(st6,i,1)="." Then Exit Do
         st5 = st5 + Mid(st6,i,1)
         i = i + 1
       Loop
       convert=st5
      End Function
```