Notes d'applications
Volume #C

"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 # 43.
Ajoutez une connexion Wifi à votre CUBLOC

Cette note d'application va vous permettre de connecter un module CUBLOC à un réseau Wifi avec une extrême simplicité. Pour ce faire on aura recours à un petit module OEM spécialisé «EZL-80C» développé par la société Sollae™. Economique, très simple à mettre en oeuvre et de petites dimensions, ce module dispose d'un connecteur prévu pour recevoir une carte 16 bits CF radio (livrée en option) au standard IEEE802.11b.

Réagissant à la manière d'un convertisseur "WLAN <> Série", le module génèrera et convertira les données qui lui seront envoyées sur son port série (niveau TTL) en un "format" TCP/IP, tout en les envoyant par radio vers le réseau local Wifi de votre PC. A l'inverse, toutes les informations en provenance du réseau radio seront restituées sur le port série du module « EZL-80C ».

Le module "EZL-80C" dispose de nombreux utilitaires de configuration et de test ainsi qu'un driver permettant de l'utiliser de façon "transparente" comme un port série virtuel côté PC (sous environnement Windows™ 2000/XP).

Notions abordées :
- Communication série

Matériel nécessaire :
- Un module « CB220 » + 4 Leds (avec résistances)
- Un module « EZL-80C » + une carte CF™ Wifi
- Un PC équipé d'une carte de communication Wifi

Description de l'application:

Cette application vous permettra depuis le programme HyperTerminal™ présent sur le PC d'envoyer **à distance et sans fil** des ordres depuis le clavier de votre PC, lesquels seront transmis au module CUBLOC via la carte Wifi de l'ordinateur. 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).

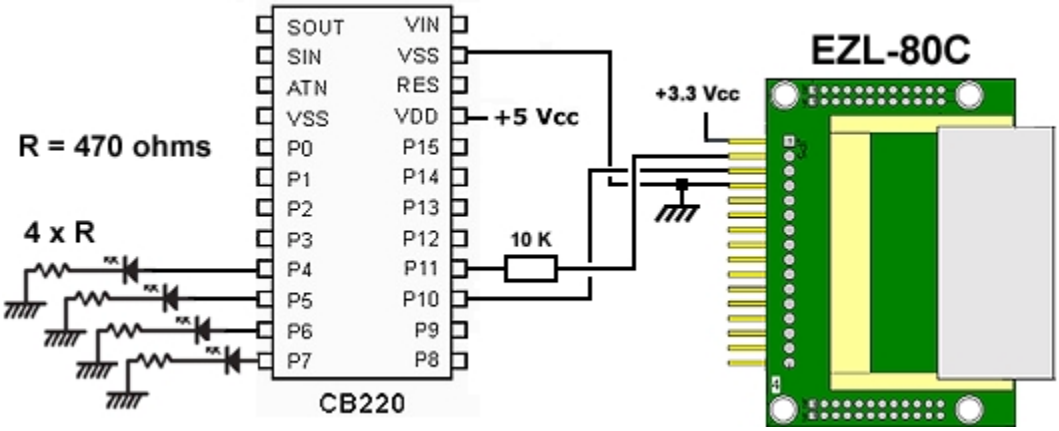
Le programme HyperTerminal™ peut également recevoir des données en provenance du module « EZL-80C » (issues du port série du CUBLOC) en les affichant à l'écran. La note d'application utilise cette possibilité pour renvoyer après chaque commande l'état des 4 sorties du CUBLOC. Ainsi après avoir sollicité 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).

Pour assurer la communication sans fil, le module CUBLOC sera connecté à un module « EZL-80C » (lequel sera équipé d'une carte CF™ de communication Wifi « WLAN/CF »).

En sollicitant la touche 0 du PC, le CUBLOC ne modifie pas l'état de ses sorties, mais renvoie 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).

Préparation matérielle:

Cette application nécessite que vous réalisiez le schéma ci-dessous.



Vous pourrez idéalement utiliser la platine de test "CB Study Board" pour réaliser le montage ci-dessus (en exploitant les Leds de celle-ci). Le module « EZL-80C » pourra également être enfiché sur la plaque de connexion sans soudure de la platine de test. Il vous faudra toutefois vous procurer un régulateur 3,3 V afin de pouvoir alimenter le module « EZL-80C ».

Si nécessaire, nous proposons une petite platine de régulation (sous la référence « REG533 »), laquelle vous permettra de générer du 3,3 V à partir du 5 V de la platine « CB Study Board »).

Les ports "P4" à "P7" seront raccordés aux Leds, tandis que les broches du port série du CUBLOC seront respectivement raccordées aux broches "TX" et "RX" du module « EZL-80C ». Une résistance de 10 Kohms permettra de limiter la tension en sortie du CUBLOC afin que cette dernière soit compatible avec le niveau max. de 3,3 V toléré par le module « EZL-80C ».

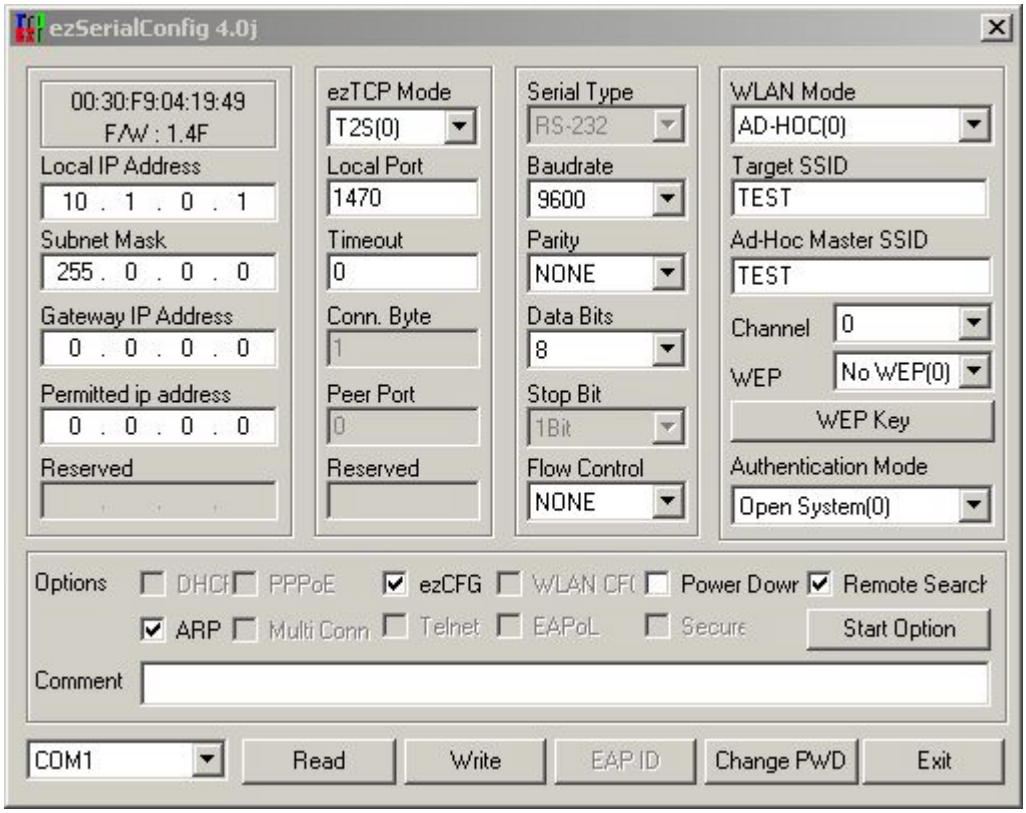
Configuration de la communication Wifi :

Le recours au module « EZL-80C » vous permettra de créer une **liaison série virtuelle** au travers de votre transmission Wifi. Une fois configuré, vous disposerez alors de l'équivalent d'un câble série RS232 (**sans fil**) entre le PC (devant être équipé d'une carte Wifi) et votre module CUBLOC (équipé du module « EZL-80C » et d'une carte CF™ Wifi). Du côté du PC, un driver de port virtuel (fonctionnant pour Windows 2000™ ou Windows XP™) permettra le pilotage automatique de la carte Wifi du PC (comme s'il s'agissait d'un port COM). Du côté CUBLOC, vous disposerez aussi de l'équivalent d'un port COM via des signaux TX et RX sur le module « EZL-80C ». Le PC et le CUBLOC pourront alors dialoguer à distance sans fil via une liaison RS-232 en passant par le réseau Wifi (comme s'ils étaient reliés physiquement par un câble série).

La première opération à réaliser consistera à configurer les paramètres du module OEM « EZL-80C » . Pour ce faire, il faudra relier temporairement ce dernier au port série de votre compatible PC (un montage d'adaptation de niveau logique à base de circuit intégré **MAX3232** (attention pas un MAX232) sera nécessaire afin que les niveaux logiques du port série du module OEM « EZL-80C » soit compatibles avec ceux du port RS232 du PC).

Si nécessaire, nous disposons d'une platine (sous la référence « EZL-90 ») réalisant cette fonction).

Une fois le module « EZL-80C » relié au port série du PC et alimenté, exécutez le programme « ezSerialConfig » (ce dernier est livré sur le CD-ROM du module « EZL-80C » (il est également disponible sur le site www.sollae.com). **Attention durant cette opération, il ne faut pas insérer la carte CF™ Wifi dans le connecteur du module « EZL-80C ».**



Sélectionnez en premier lieu le N° du port COM sur lequel est relié le module « EZL-80C », puis cliquez sur le bouton « Read » pour récupérer la configuration du module « EZL-80C ». A ce stade, vous pouvez modifier les paramètres du module selon votre application.

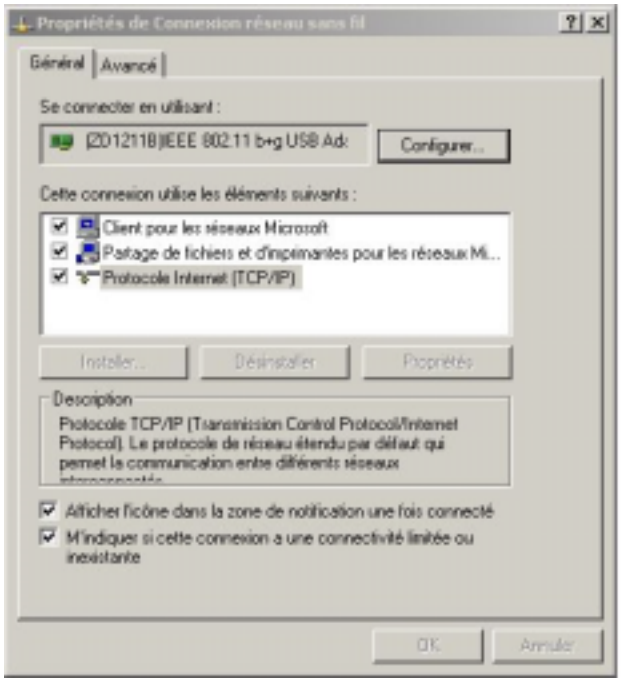
Dans notre exemple le module est destiné à dialoguer directement avec un PC équipé d'une carte de communication Wifi et sans cryptage. On devra donc configurer le paramètre « **WLAN Mode** » en « **AD-HOC(0)** ».

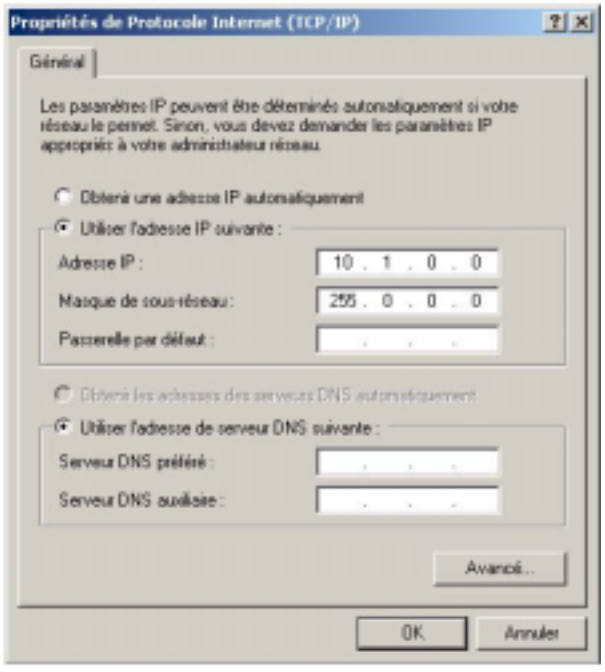
Recopiez les autres paramètres présent la copie d'écran de la page précédente sur la fenêtre d'« ezSerialConfig » (certains de ces paramètres peuvent nécessiter des modifications si vous disposez d'un environnement Wifi différent).

Par contre, pour les besoins de notre application il faut impérativement configurer le débit du port à 9600 bds / sans parité / 8 bits de données et sans contrôle de flux. Terminer l'opération en cliquant sur le bouton « **Write** ».

Vous pouvez désormais déconnecter le module « EZL-80C » du PC et (hors alimentation) insérer la carte CF™ Wifi dans le connecteur du module
« EZL-80C », lequel pourra alors être relié au module PICBASIC comme indiqué ci-avant.

Sur le PC équipé de la carte de communication Wifi, Sous Windows™ allez dans le menu « **Démarrer** » -> « **Paramètres** » -> « **Connexions réseau** ». Une fenêtre s'affiche alors dans laquelle il vous faut réaliser un clic droit de souris sur l'icône « **Connexion réseau sans fil** » et sélectionnez « **Propriétés** » (L'écran de gauche présenté ci-dessous s'affiche alors).





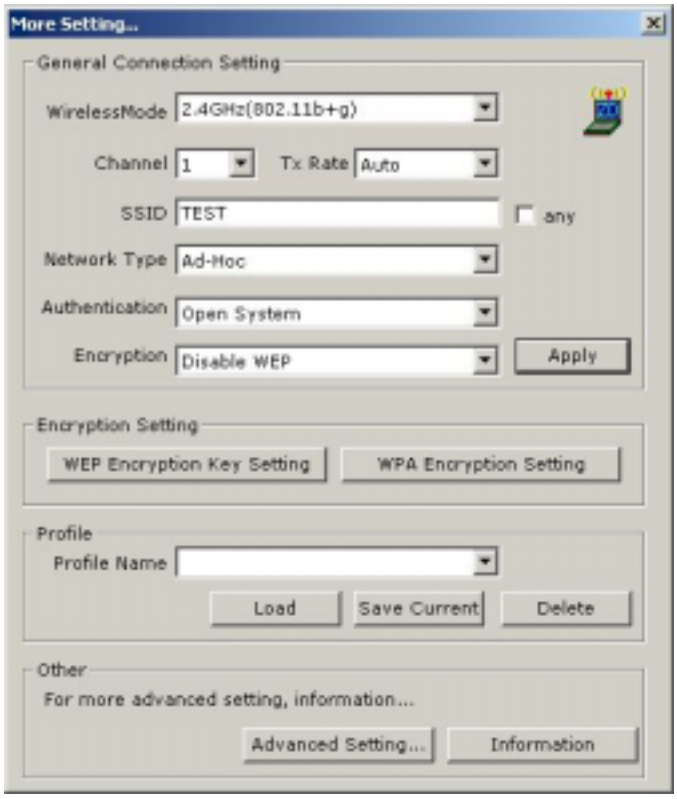
Faite un « double-clic » sur « **Protocole Internet (TCP/IP)** » afin d'afficher la seconde fenêtre de droite présentée ci-dessus.

Cochez la sélection « **Utiliser l'adresse IP suivante** », puis saisissez une adresse IP pour la carte wifi du PC (différente de celle utilisée pour la carte « EZL-80C ») ainsi que la valeur du masque « sous réseau » (qui doit être identique à celle utilisée lors de la configuration de la carte « EZL-80C »). Cliquez ensuite sur le bouton « OK ».

Exécutez ensuite le logiciel de configuration de la carte Wifi de votre PC et modifiez les paramètres de la communication afin que ces derniers correspondent à ceux programmés sur le module « EZL-80C ».

Dans notre exemple, on sélectionnera le même nom d'identification « TEST » que celui choisi dans « ezSerial Config » pour le paramètre SSID.

De même on sélectionnera un mode de communication Ad-Hoc, sans cryptage.



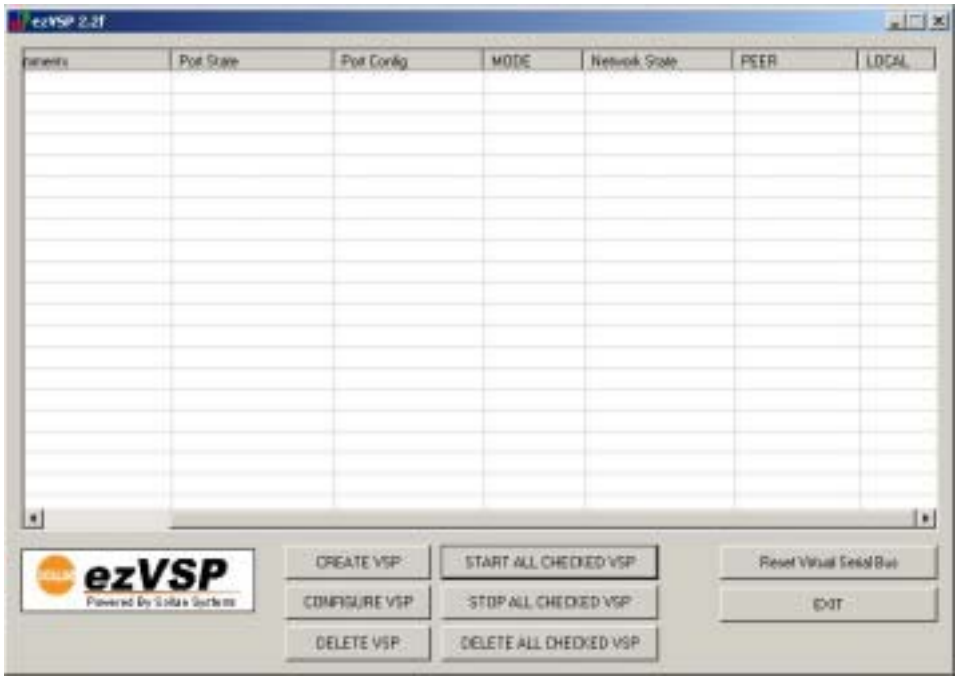
Cet écran de configuration peut être différent selon le type de carte Wifi que vous utilisez.

De même, certains de ces paramètres peuvent nécessiter des modifications si vous disposez d'un environnement Wifi différent.

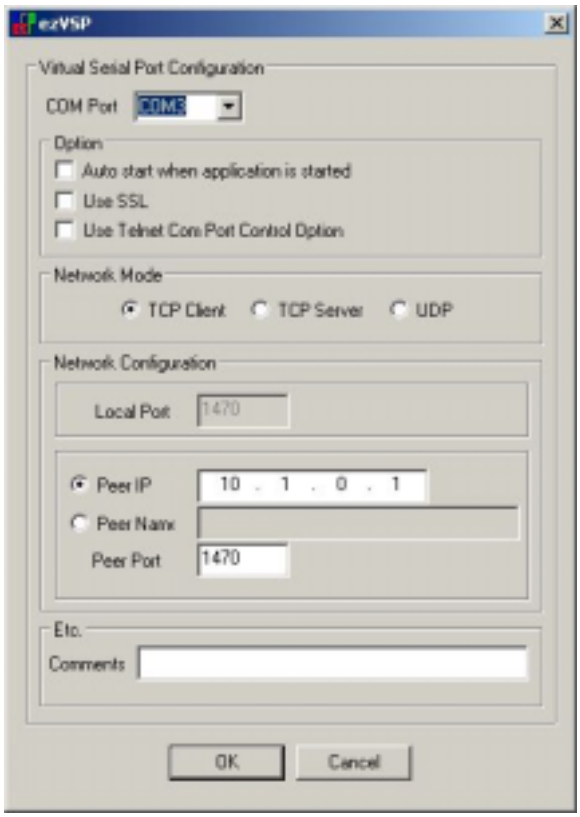
La dernière opération consiste à créer un port de communication virtuel. Pour ce faire, il vous faudra télécharger un driver (**ezVSP**) sur le site www.sollae.com. **Attention ce driver ne fonctionne que sous WindowsXP/2000™.**

Le site demande que vous lui communiquiez l'adresse MAC inscrite sur votre module « EZL-80C » pour pouvoir installer le driver (N'indiquez pas l'adresse MAC de la carte CF™ Wifi).

Une fois installé, exécutez « ezVSP » afin d'obtenir l'écran ci-dessous :



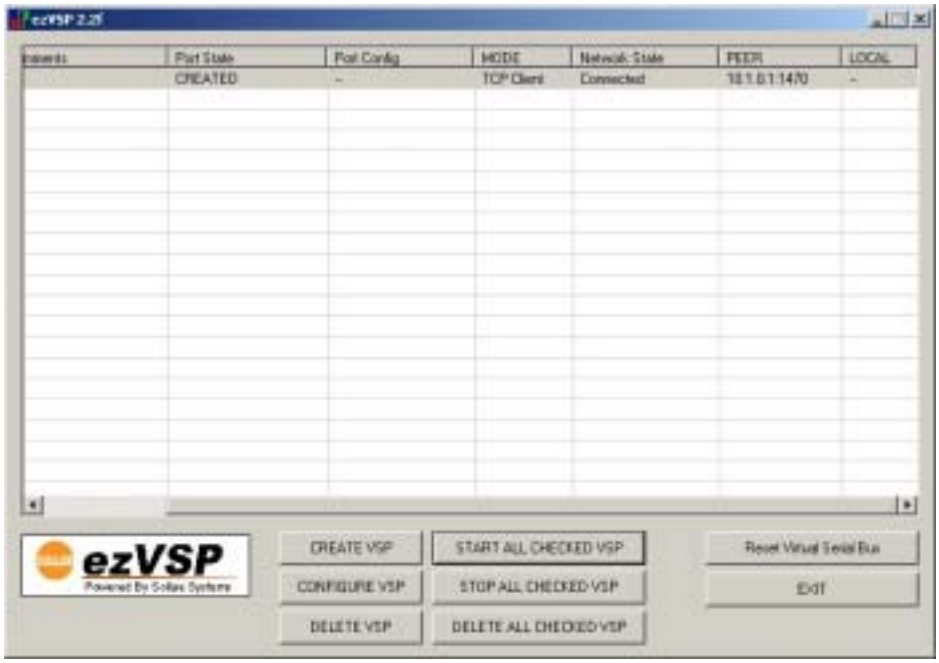
Cliquez ensuite sur le bouton « **CREATE VSP** » afin que la fenêtre ci-dessous s'ouvre.



Sélectionnez le N° du port COM virtuel qui sera créé.

Dans la sélection « **Network mode** », cochez l'option « **TCP Client** ». Configurez l'adresse « **Peer IP** » avec celle du module « EZL-80C » et le paramètre « Peer Port » avec la même valeur que celle utilisée pour le module «EZL-80C ».

Cliquez enfin sur le bouton « OK » pour revenir à l'écran ci-dessous



Cliquez alors sur le bouton « **START ALL CHECKED VSP** » afin de créer définitivement le port virtuel. Votre PC est désormais prêt à communiquer avec la carte « EZL-80C » afin que vous puissiez mettre en œuvre cette application.

Interprétation du programme : (Ce dernier est présent sur notre site Internet)

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 (lequel reçoit ses infos de la part de la sortie TX du module « EZL-80C »). 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 en série au module « EZL-80C » (lequel trasnmet à son tour ces infos à la carte Wifi du PC) afin que ces phrases s'affichent dans la fenêtre de HyperTerminal™ de l'ordinateur.

A noter que ce programme est très simple et ne comprend aucune sécurisation lors de l'activation des sorties. Il sera très aisé de le perfectionner en ajoutant quelques octets en entête de la commande afin que le CUBLOC analyse ces octets et ne réagisse que s'il les « reconnaît ».

```
'####################################
'#        Ajoutez une connexion Wifi        #
'#          à votre module CUBLOC           #
'#     @Lextronic 2007 - 05/12/2007         #
'####################################

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.",13,10
    Else
      Putstr 1,"La sortie N° 1 est active.",13,10
    End If
    Delay 100
    If sortie2=0 Then
      Putstr 1,"La sortie N° 2 est non active.",13,10
    Else
      Putstr 1,"La sortie N° 2 est active.",13,10
    End If
    Delay 100
    If sortie3=0 Then
      Putstr 1,"La sortie N° 3 est non active.",13,10
    Else
      Putstr 1,"La sortie N° 3 est active.",13,10
    End If
    Delay 100
    If sortie4=0 Then
      Putstr 1,"La sortie N° 4 est non active.",13,10,10
    Else
      Putstr 1,"La sortie N° 4 est active.",13,10,10
    End If
   End If
  Loop
```

NOTE D'APPLICATION # 44.
Ajoutez une connexion Bluetooth™ à votre CUBLOC

Cette note d'application va vous permettre réaliser une liaison série sans fil entre un compatible PC et un module CUBLOC au moyen d'un dispositif de communication Bluetooth™. Côté PC, on utilisera une clef USB Bluetooth™ (si votre PC n'est Pas déjà pré-équipé d'un dispositif pré-intégré). Côté CUBLOC, On utilisera un module spécialisé « F2M03GLA ».

Le "F2M03GLA" est un module hybride subminiature Bluetooth™ v2+EDR pré-qualifié faible consommation. Doté d'une antenne intégrée et d'une puissance d'émission de +8 dBm, il dispose d'une portée maximale de l'ordre de 250 m en terrain dégagé (s'il est utilisé avec un module de même catégorie).

Extrêmement compact (28,5 x 15,2 x 2 mm), performant et économique, le module hybride "F2M03GLA" est de part sa puissance améliorée, sa "bonne" sensibilité (-83 dBm) et sa faible consommation, tout naturellement destiné à être intégré au sein d'applications embarquées les plus diverses. Le Firmware de base permet au module de supporter le protocole de communication **SPP** (Serial port Profile). Avec ce protocole, toutes les informations arrivant sur le port série du "F2M03GLA" seront automatiquement transférées de façon transparente au périphérique connecté sur la liaison Bluetooth™. La communication (bien évidément bi-directionnelle) s'apparente alors à un véritable câble série virtuel sans fil.

Le module "F2M03GLA" dispose de connexions présentes au dos du circuit. Il est possible de se raccorder directement sur ces dernières via des fils très fin (type fils à wrapper par exemple). Dans notre cas, nous avons préféré utiliser une petite platine d'interface spécialement conçue pour faciliter le câblage du « F2M03GLA ». Cette platine (en plus d'intégrer un régulateur 3,3 V et un circuit RESET) reprend les différents signaux du module Bluetooth™ sur des broches au pas standard de 2,54 mm.

Notions abordées :
- Communication série

Matériel nécessaire :
- Un module « CB220 » + 4 Leds (avec résistances)
- Un module « F2M03GLA » + circuit d'adaptation « F2M-AD1 »
- Un PC équipé d'une communication Bluetooth™

Description de l'application:

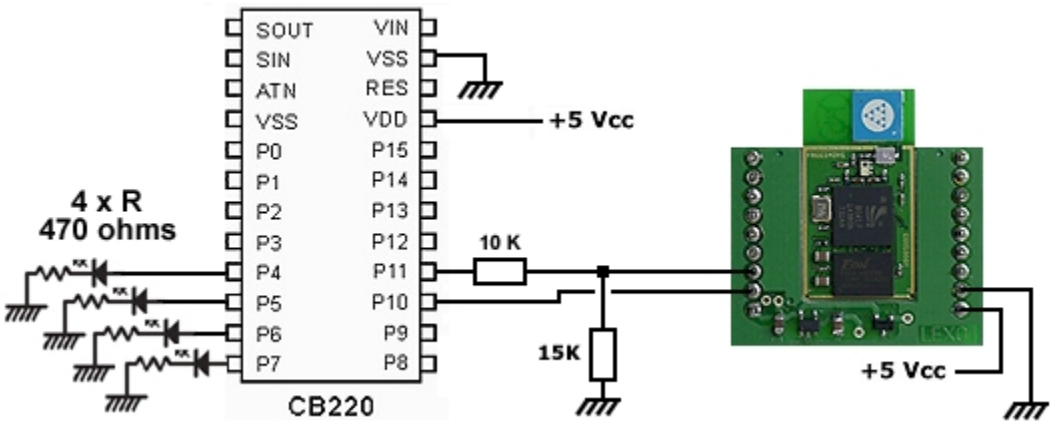
Cette application vous permettra depuis le programme HyperTerminal™ présent sur le PC d'envoyer **à distance et sans fil** des ordres depuis le clavier de votre PC, lesquels seront transmis au module CUBLOC via la communication Bluetooth™ de l'ordinateur. Ainsi en sollicitant les touches 1 à 4 du PC, 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 module « F2M03GLA » (issues du port série du CUBLOC) 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 pas l'état de ses sorties, mais renvoie 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).

Préparation matérielle:

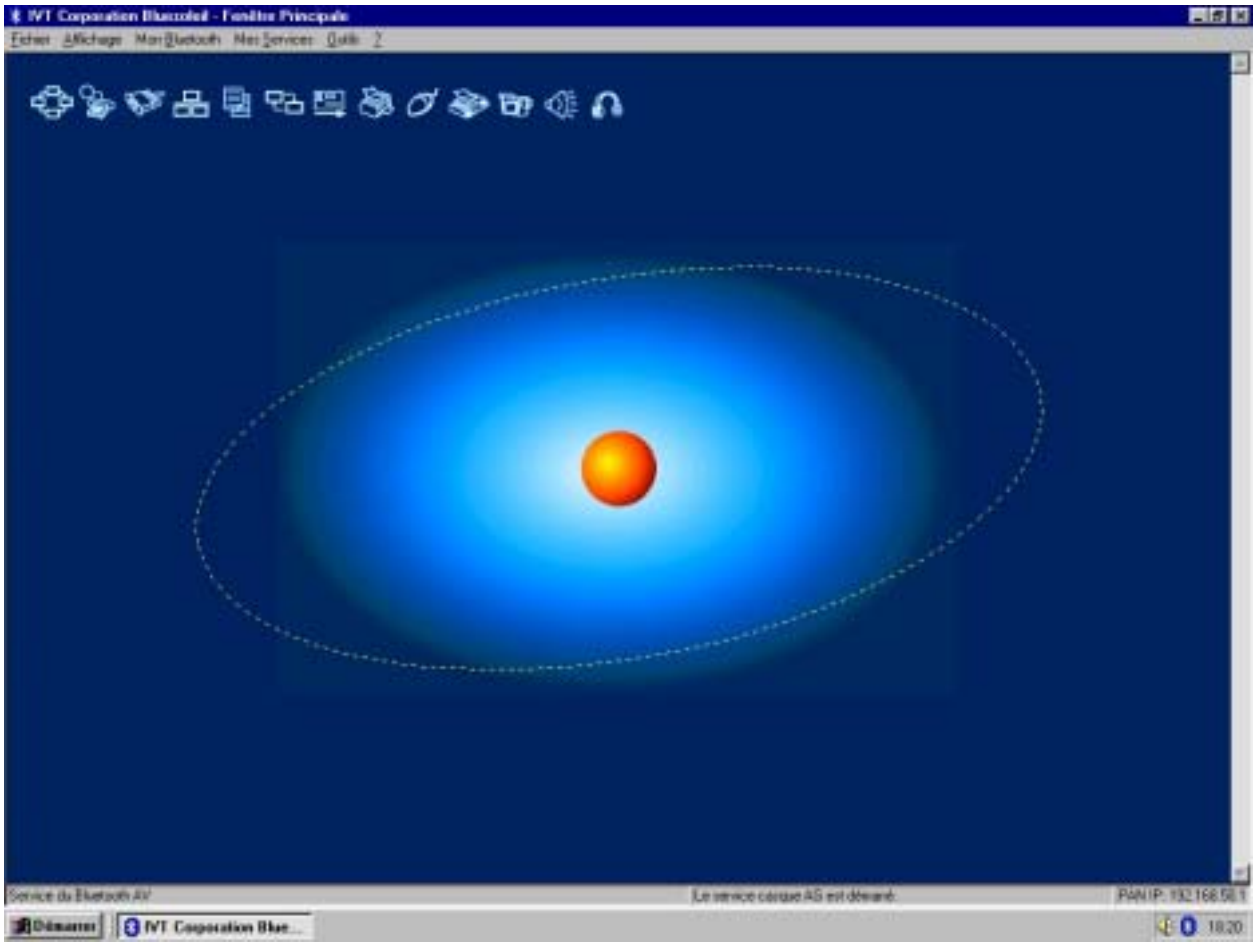
Cette application nécessite que vous réalisiez le schéma ci-dessous.



Vous pourrez idéalement utiliser la platine de test "CB Study Board" pour réaliser le montage ci-dessus (en exploitant les Leds de celle-ci). Les ports "P4" à "P7" seront raccordés aux Leds, tandis que les broches du port série du CUBLOC seront respectivement raccordées aux broches "TX" et "RX" du module « F2M03GLA ». Un pont de résistance permettra de limiter la tension en sortie du CUBLOC afin que cette dernière soit compatible avec le niveau max. de 3,3 V toléré par le module « F2M03GLA ».

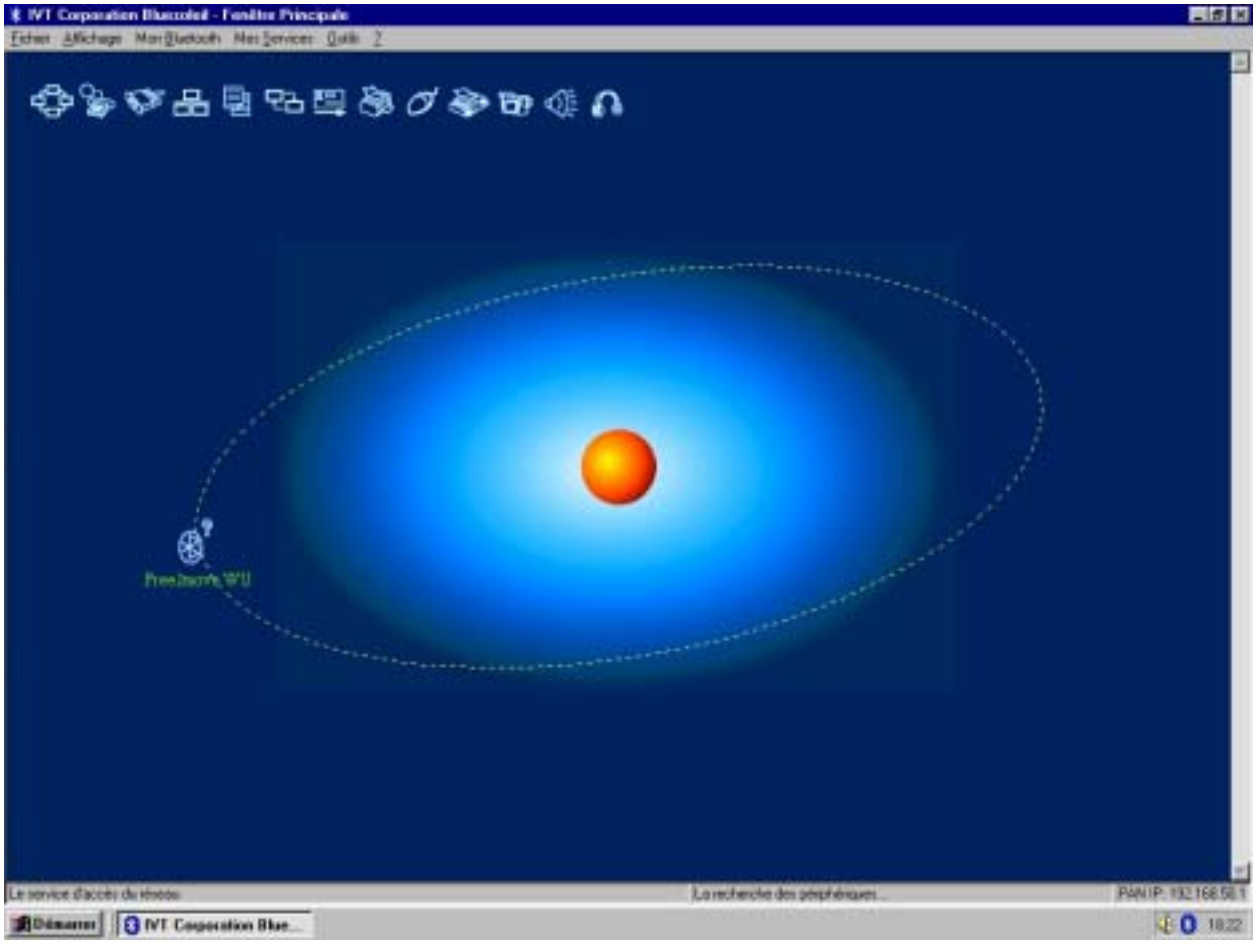
Configuration de la communication Bluetooth™ :

A la mise sous tension du module « F2M03GLA », ce dernier est prè-configuré en mode « esclave » et attend de recevoir un ordre de connexion de la part d'un périphérique Bluetooth™ maître pour initier une communication série sans fil transparente. Pour ce faire « lancez » le logiciel de configuration du périphérique Bluetooth™ de votre PC.



Votre logiciel peut être différent de l'écran présenté ci-dessus.

Effectuez alors une recherche de périphériques Bluetooth™



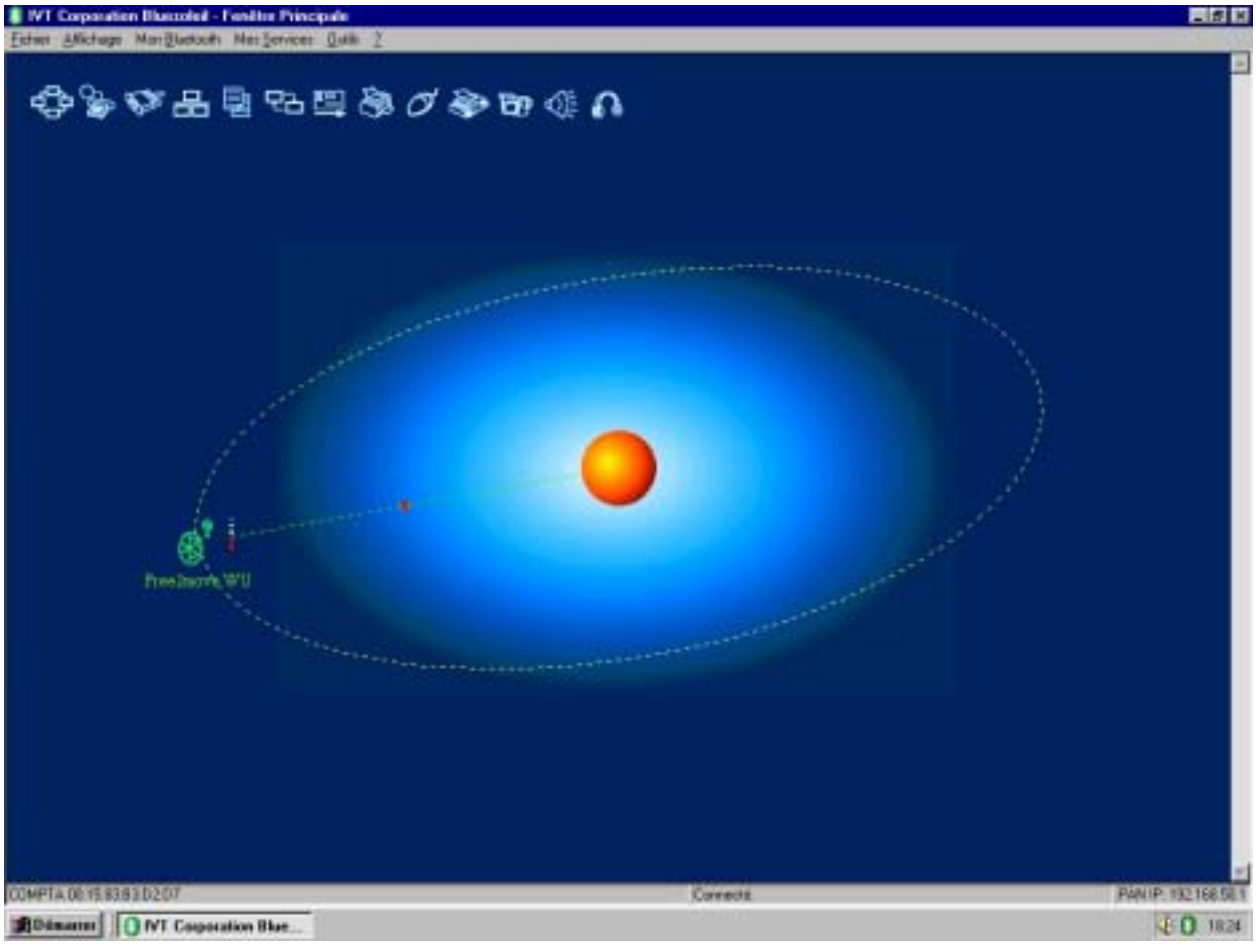
A ce stade, votre PC doit détecter et afficher les modules Bluetooth™ présent dans l'aire de sa couverture radio. Dès lors vous devez voir apparaître le module Free2move sur votre logiciel.

Si ce dernier n'apparaît pas, vérifiez votre cablage ainsi que la tension d'alimentation de votre montage.

Si le périphérique Free2move est correctement détecté, votre logiciel reconnaîtra la nature du protocole SPP du module « F2M03GLA » et il vous sera possible dans les options du programme de gestion Bluetooth™ de votre PC d'établir une connexion « série » via un port COM virtuel (exemple port COM6).

Dès lors toutes les données envoyées depuis le PC vers sur le port COM6 seront transmises via le péripérique Bluetooth™ du PC vers le module « F2M03GLA » et disponibles sur la sortie RX du module « F2M03GLA ».

A l'inverse toutes les données séries envoyées vers le module « F2M03GLA » (via le CUBLOC) seront réceptionnées via le périphérique Bluetooth™ du PC et prises en compte en tant que données séries sur le port COM6.



Il ne reste plus alors qu'à ouvrir une cession hyperterminal (en sélectionnant le N° du port COM virtuel créé par le programme de gestion Bluetooth™) afin de pouvoir faire communiquer le PC et l'application à base de module CUBLOC + « F2M03GLA ».

Configurez la vitesse de communication à 38400 bds / 8 bits / pas de parité / 1 bit de stop et aucun contrôle de flux.

Attention : certaines versions « anciennes » d'Hyperterminal (notamment sous Windows 98™) ne permettent pas toujours de pouvoir sélectionner de port COM virtuel.

Côté CUBLOC, il vous faudra charger exactement le même programme que celui de la note d'application N° 43 (voir page 13). Toutefois il vous faudra changer la ligne du programme :

Opencom 1,9600,3,10,35

Par

Opencom 1,38400,3,10,35

Afin de configurer le port série du CUBLOC à 38400 bds.

Le mode de fonctionnement et les explications du programme proprement dits sont alors en tout point identiques à ceux indiqués dans la note d'application précédente (voir « Interprétation du programme page 12) – la seule différence est que la communication sans fil Wifi utilisée précédemment est remplacée par une communication Bluetooth™.

Consultez le lien suivant pour obtenir plus d'infos sur les modules Bluetooth™ Free2move :

http://www.lextronic.fr/R1317-free2move.html

Bluetooth™ is a trademark owned by Bluetooth SIG, inc, U.S.A

NOTE D'APPLICATION # 45.
Jeux vidéos type « casse brique »

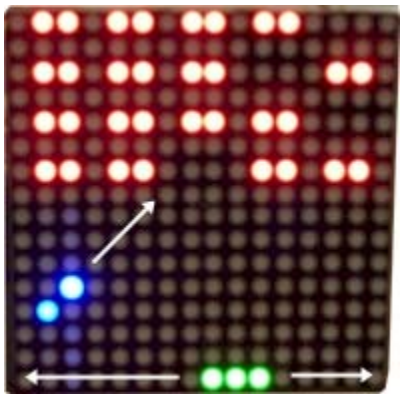
Cette note d'application va vous permettre de réaliser un jeu vidéo type « casse brique » à l'aide d'un CUBLOC CB220B associé à une ou plusieurs matrices à Leds multicolores Sparkfun™. Ces matrices se composent de 64 Leds pouvant indépendamment être allumées avec 7 couleurs différentes. Les matrices se pilotent par le biais d'une liaison SPI™. 2 variantes de jeu de « casse brique » seront proposées. La première utilisera une seule matrice à Leds et la seconde mettra en œuvre 4 matrices.

Le pilotage des matrices est très simple. Il vous suffit simplement de leur envoyer une suite de 64 octets (correspondant aux 64 Leds de la matrice). Chaque octet pourra avoir une valeur comprise entre 0 et 7 (ces valeurs correspondent à la couleur que devra prendre la leds : 0 = éteinte, 1 = rouge, 2 = vert, etc)

Notions abordées :
- Gestion de variable de type tableau
- Communication de type SPI™

Matériel nécessaire :
- Un module « CB220 » + 1 ou 4 matrices à Leds multicolores Sparkfun™

Description de l'application:

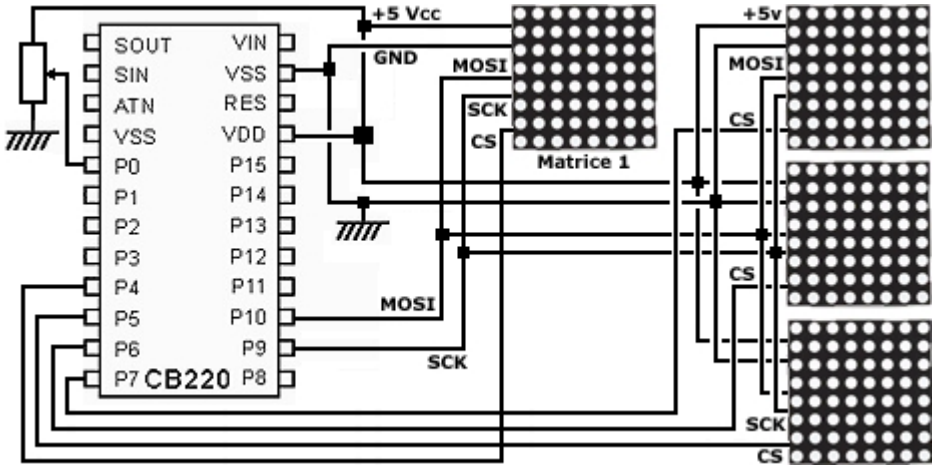


Le jeu de « casse brique » se présente sous la forme d'un écran (matérialisé par les matrices à Leds) sur lequel des « briques » sont disposées sur sa partie haute. Parallèlement à ces briques, une balle se déplace sur l'écran. Le but du jeu consiste à faire en sorte que la balle entre en collision avec toutes les briques de l'écran afin de les faire disparaître. Pour ce faire, vous déplacer une « raquette » latéralement au bas de l'écran afin de pouvoir renvoyer la balle vers le haut. Vous disposez de 3 « vies » pour parvenir à détruire le plus possible de brique. Vous perdez une vie à chaque fois que vous ne parvenez pas à renvoyer la balle et que cette dernière dépasse la partie inférieur de l'écran. Si vous réussissiez à détruire toutes les briques présentent à l'écran, vous passez au niveau supérieur et la vitesse de déplacement de la balle accélère.

Pour la réalisation de cette aplication, nous avons utilisé une platine de développement « CUBLOC Study Board » dont un des potentiomètres sera mis à profit pour assurer le déplacement de la raquette. Tous les éléments de l'écran auront des couleurs différentes afin de pouvoir facilement les identifier (la raquette en vert, la balle en bleu, les briques en rouge…).

Préparation matérielle:

Cette application nécessite que vous réalisiez le schéma ci-dessous.



Si vous n'utilisez qu'une seule matrice, il vous suffit simplement de ne pas tenir compte des connexions des matrices 2 à 4. Les broches MISO des matrices à Leds ne seront pas utilisée et devront restées libres.

Le port P0 est utilisé en entrée de conversion « analogique/numérique » afin de venir « lire » la position du potentiomètre. Les ports P4 à P7 servent à piloter les entrées de sélection des différentes matrices. Les ports P9 et P10 servent pour la communication SPI™.

Interprétation du programme : (Ce dernier est présent sur notre site Internet)

Le programme ci-dessous concerne la réalisation du jeu de « casse brique » en version 1 matrice. Ce dernier est largement commenté et ne devrait pas poser trop de problème de compréhension. Le principe de fonctionnement repose sur l'utilisation d'une variable de type tableau à 2 dimensions qui s'apparente à une « image » de la matrice. Ce tableau renferme ainsi la position et l'état de chaque Led de la matrice, mais également la position de la raquette et la position de la balle. Le tableau est remis à jour avant chaque rafraîchissement de l'affichage de la matrice à Leds. Le début de la partie se matérialise par un « balayage furtif orange » de l'écran, suivi par l'affichage des briques, de la raquette et de la balle. La balle reste immobile tant que vous ne déplacez pas le potentiomètre de la raquette. Lorsque le programme détecte un déplacement de cette dernière, la balle commence à bouger. 2 autres variables sont attribuées aux déplacements horizontaux et verticaux de la balle.

Une série de tests permettent de détecter les collisions entre la balle et les briques, entre la balle et la raquette ou entre la balle et les bords de l'aire de jeu. Dès lors différentes actions seront déclenchées en fonction de ces évènements : effacement d'une brique, rebond de la balle avec changement de direction, perte d'une vie, etc… A noter que la balle aura différentes réactions de rebond selon que vous « retourner » celle-ci sur le bord ou le centre de la raquette. Le programme totalise le nombre de brique détruite ainsi que le nombre de vie restante. Lorsque vous perdez toutes vos vies, le programme recommence son exécution depuis le début de la partie.

Lorsque toutes les briques sont détruites, une flèche bleue apparaît à l'écran pour signaler que vous changez de niveau (la vitesse de jeu augmente alors).

La version du programme mettant en œuvre 4 matrices est très similaire mise à part que les limites de détection de l'aire de jeu seront différentes et qu'il faudra gérer une variable tableau avec plus d'éléments et rafraîchir 4 matrices au lieu d'une seule.

Améliorations possibles :

Nous vous laissons le soin d'apporter des améliorations au programme en ajoutant par exemple des sons, un score, en modifiant le nombre et la position initiale des briques en fonction du niveau, etc…

```
'#########################################################################
'# Casse brique sur 1 matrice a leds multi-couleurs Sparfun
'# @Lextronic 2008 - 19/10/2008 - Realise avec CUBLOC Studio Version 2.5.
'#########################################################################

 Const Device = CB220

 Dim mat1(9,9) As Byte                                  ' Memoire RAM matrice 1
 Dim x As Byte
 Dim y As Byte
 Dim xballe As Byte                                     ' Coordonnees balle
 Dim yballe As Byte
 Dim dirxballe As Long                                  ' Direction x balle
 Dim Diryballe As Long                                  ' Direction y balle
 Dim vitballe As Integer                                ' Vitesse deplacement balle
 Dim difficulte As Integer                              ' Difficulte du jeu
 Dim brique As Byte                                     ' Nb Brique
Dim posraq As Byte
Dim raquette As Integer                                 ' Potentiometre de la raquette
Dim action As Byte                                      ' Variable gestion balle
Dim vie As Byte                                         ' Nombre de vie

High 4                                                  ' Port 4 en sortie pilotage CS matrice 1 = 1
Input 0                                                 ' Port 0 entree - potentiometre raquette
difficulte = 10                                         ' Initialise vitesse du jeu
vie = 3                                                 ' Initialise nombre de vies

'-------- Ecran de presentation et affichage des briques ----------

For y = 1 To 8                                          ' Efface la matrice
  For x = 1 To 8
    mat1(x,y)=0
  Next
Next
Gosub  affmat1                                          ' Raffraichissement de la matrice

For yballe = 1 To 8                                     ' Affiche matrice en orange
   For xballe = 1 To 8
     mat1(xballe,yballe)=4
   Next
  Gosub  affmat1                                        ' Raffraichissement de la matrice
Next

Delay 100
Gosub initbrique                                        ' Affiche les briques
Gosub depballe                                          ' Gestion deplacement balle
```

```
 Gosub  affmat1                                                    ' Raffraichissement de la matrice

'-------------------- Attend deplacement raquette pour debuter le jeu ---------------------

raquette=Adin(0)
vitballe = raquette                                               ' Memorise position raquette
Do
   raquette=Adin(0)
   If Abs(raquette - vitballe) > 120  Then Exit Do
Loop
vitballe = difficulte                                             ' Initialise tempo vitesse deplacement balle

Do
    Gosub depballe                                                ' Gestion deplacement balle

 '-------------------- Gestion deplacement de la balle ----------------------

   vitballe = vitballe-1
   If vitballe = 0 Then
      action = 0
      vitballe = difficulte                                       ' Reinitialise vitesse du jeu

'----------------- Test si loupe la raquette ------------------
    If action = 0 And yballe = 8 Then action = 5

'----------------- Test collision raquette --------------------
    If action = 0 And yballe+Diryballe = 8 Then
       If dirxballe = 1 Then
       If xballe+dirxballe = posraq Then action = 4
       If xballe+dirxballe = posraq+1 Then action = 3
       If xballe+dirxballe = posraq+2 Then action = 3
    Else
       If xballe+dirxballe = posraq Then action = 3
       If xballe+dirxballe = posraq+1 Then action = 3
       If xballe+dirxballe = posraq+2 Then action = 4
    Endif
Endif

'----------------- Test depassement droite/haut ---------------------
   If action = 0 And xballe+dirxballe > 8 And yballe+Diryballe < 1 Then action = 4

'----------------- Test depassement gauche/haut ---------------------
   If action = 0 And xballe+dirxballe < 1 And yballe+Diryballe < 1 Then action = 4

'----------------- Test depassement bord droit ----------------------
   If action = 0 And xballe+dirxballe > 8 Then action = 2

'----------------- Test depassement bord gauche ---------------------
   If action = 0 And xballe+dirxballe < 1 Then action = 6

'----------------- Test depassement bord haut ----------------------
   If action = 0 And yballe+Diryballe < 1 Then action = 3

'----------------- Test si collision avec une brique ----------------------
  If action = 0 And mat1(xballe+dirxballe,yballe+Diryballe) <> 0 Then action = 1

mat1(xballe,yballe) = 0                                           ' Efface ancienne position de la balle
```

```
Select Case action
        Case 0                                          ' Deplacement normal de la balle
          Gosub affballe                                ' Affiche la balle

        Case 1                                          ' Collision avec une brique
          brique = brique -1                            ' Une brique en moins !
          xballe = xballe+dirxballe                     ' Reactualise position x balle
          yballe = yballe+Diryballe                     ' Reactualise position y balle
          mat1(xballe,yballe) = 3                       ' Affiche balle
          mat1(xballe+1,yballe) = 0                     ' Efface brique
          mat1(xballe-1,yballe) = 0                     ' Efface brique
          Diryballe=Diryballe*-1                        ' La balle change de direction

        Case 2                                          ' Detection bord droit de la matrice
          dirxballe=dirxballe*-1                        ' La balle change de direction
          If mat1(xballe+dirxballe,yballe+Diryballe) <> 0 Then
             brique = brique -1                         ' Une brique en moins !
             xballe = xballe+dirxballe                  ' Reactualise position x balle
             yballe = yballe+Diryballe                  ' Reactualise position y balle
             mat1(xballe,yballe) = 0                    ' Affiche balle
             mat1(xballe+1,yballe) = 0                  ' Efface brique
             mat1(xballe-1,yballe) = 0                  ' Efface brique
             Diryballe=Diryballe*-1                     ' La balle change de direction
          Endif
          Gosub affballe                                ' Affiche la balle

        Case 3                                          ' Detection bord haut de la matrice
          Diryballe=Diryballe*-1                        ' La balle change de direction
          Gosub affballe                                ' Affiche la balle

        Case 4                                          ' Detection collision bord de la raquette
          dirxballe=dirxballe*-1                        ' La balle change de direction
          Diryballe=Diryballe*-1                        ' La balle change de direction
          Gosub affballe                                ' Affiche la balle

        Case 5                                          ' Detection perte de la balle
           For x = 1 To 8
               mat1(x,8)=7
          Next
          Gosub  affmat1                                ' Raffraichissement de la matrice
          Delay 400
          vie = vie - 1                                 ' On perd une vie
          Gosub  initballe                              ' Initialise position initiale de la balle

        Case 6                                          ' Detection bord gauche de la matrice
          dirxballe=dirxballe*-1                        ' La balle change de direction
         If yballe+Diryballe = 8 And mat1(xballe+dirxballe,yballe+Diryballe)<> 0 Then Diryballe=Diryballe*-1
         Gosub affballe

      End Select

      '----------------- Test passage niveau superieur ? ---------------------

      If brique = 0 Then

        For yballe = 1 To 8                     ' Effacement de la matrice
            For xballe = 1 To 8
              mat1(xballe,yballe)=0
            Next
        Next
```

```
            For yballe = 1 To 8                    ' Memorise fleche dans la matrice
                For xballe = 4 To 5
                  mat1(xballe,yballe)=5
                Next
              Next
              For xballe = 2 To 7
                  mat1(xballe,3)=5
              Next
              mat1(3,2)=5
              mat1(6,2)=5
              Gosub  affmat1                   ' Raffraichissement de la matrice
              Delay 1000
              Gosub initbrique                 ' affiche a nouveau les briques
              difficulte = difficulte -1
              If difficulte = 0 Then difficulte = 1
            Endif

           '------------------ Test si fin de partie ? --------------------
             If vie = 0 Then                   ' Fin de partie ?
                  Delay 200
                  Reset
            Endif

      Endif

      Gosub  affmat1                   ' Raffraichissement de la matrice

   Loop

           '---------------------- Rafraichissement de la matrice -----------------------
affmat1:
           Low 4                       ' Selection CS matrice 1
           For y = 1 To 8
              For x = 1 To 8
                Shiftout 9,10,1,mat1(x,y),8
             Next
           Next
           High 4                                    ' Desactivation CS matrice 1
    Return

           '---------------------- Rafraichissement affichage balle -----------------------
affballe:
     xballe = xballe+dirxballe                       ' Reactualise position x balle
     yballe = yballe+Diryballe                       ' Reactualise position y balle
     mat1(xballe,yballe) = 3                         ' Affiche nouvelle position de la balle

     Return

          '---------------------- Initialisation position initiale de la balle -----------------------
initballe:
      xballe = 3                                     ' Initialise position initiale de la balle
      yballe = 7
      mat1(xballe,yballe)=3                          ' memorise la balle dans la matrice
     dirxballe=1                                     ' La balle remonte en haut doite a 45 degre
     Diryballe=-1
     Return
```

```
          '-------------------- Gestion deplacement de la raquette ---------------------
depballe:
           raquette=Adin(0)
           If raquette < 300 Then raquette = 300        ' limite position gauche raquette
           If raquette > 900 Then raquette = 900
           raquette = raquette - 300
           posraq=(raquette/120)+1                      ' Calcul position raquette
           For x = 1 To 8                               ' Memorisation raquette dans la matrice
             mat1(x,8)=0
           Next
           mat1(posraq,8)=2
           mat1(posraq+1,8)=2
           mat1(posraq+2,8)=2
          Return

                   '-------------------- Initialisation briques sur la matrice ---------------------
initbrique:
          For yballe = 1 To 8                           ' effacement de la matrice
              For xballe = 1 To 8
                mat1(xballe,yballe)=0
              Next
             Gosub  affmat1                             ' Raffraichissement de la matrice
           Next

           For y = 1 To 3 Step 2                        ' Placement des briques
               For x = 1 To 7 Step 3
                   mat1 (x,y)=1
                  mat1 (x+1,y)=1
              Next
           Next
           brique = 6                                   ' Initialise nombre de briques initiale
           Gosub  initballe                             ' Initialise position initiale de la balle
           Return
```

NOTE D'APPLICATION # 46.
Jeux vidéos type « Serpent »

Cette note d'application va vous permettre de réaliser un jeu vidéo type « Serpent » à l'aide d'un CUBLOC CB220B associé à une ou plusieurs matrices à Leds multicolores Sparkfun™. Ces matrices se composent de 64 Leds pouvant indépendamment être allumées avec 7 couleurs différentes. Les matrices se pilotent par le biais d'une liaison SPI™. 2 variantes de jeu de « casse brique » seront proposées. La première utilisera une seule matrice à Leds et la seconde mettra en œuvre 4 matrices.

Le pilotage des matrices est très simple. Il vous suffit simplement de leur envoyer une suite de 64 octets (correspondant aux 64 Leds de la matrice). Chaque octet pourra avoir une valeur comprise entre 0 et 7 (ces valeurs correspondent à la couleur que devra prendre la leds : 0 = éteinte, 1 = rouge, 2 = vert, etc)

Notions abordées :
- Gestion de variable de type tableau
- Communication de type SPI™

Matériel nécessaire :
- Un module « CB220 » + 1 ou 4 matrices à Leds multicolores Sparkfun™

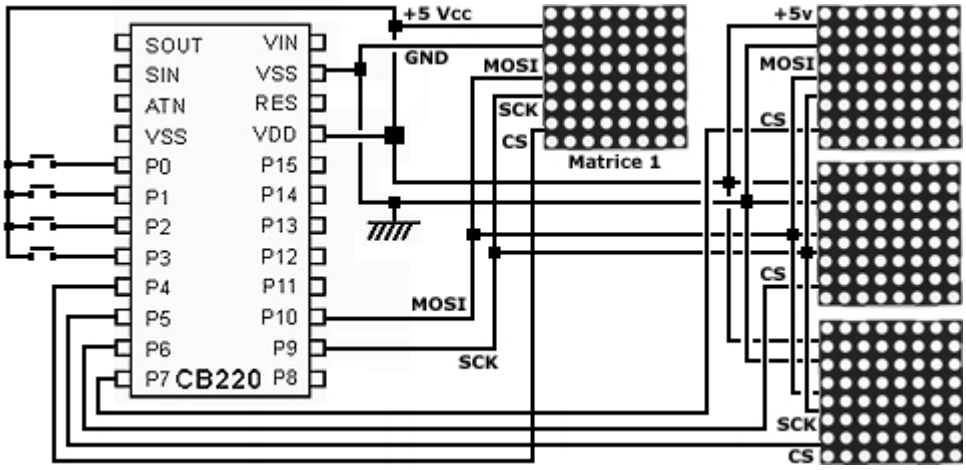
Description de l'application:

Le jeu du « serpent » se présente sous la forme d'un consiste à déplacer un serpent sur l'écran à l'aide de 4 touches (haut/bas/droite/gauche). Des pommes de couleurs aléatoires apparaissent à l'écran. Lorsque le serpent mange les pommes rouges, il grandit d'un anneau. les pommes bleues foncés enlèvent un anneau, les pommes bleues clair enlèvent 3 anneaux, les pommes oranges accelèrent le déplacement du serpent, les pommes violettes le ralentisse, les pommes blanches sont mortelles. Le but du jeu est de rester le plus longtemps et d'être le serpent le plus long possible.
Si le serpent se recoupe (mange un de ses anneaux) ou sort de l'aire de jeu (ou mange une pomme blanche), la partie est finie..

Pour la réalisation de cette aplication, nous avons utilisé une platine de développement « CUBLOC Study Board » dont 4 boutons-poussoirs seront mis à profit pour assurer le déplacement du serpent (haut / bas / droite / gauche).

Préparation matérielle:

Cette application nécessite que vous réalisiez le schéma ci-dessous.



Si vous n'utilisez qu'une seule matrice, il vous suffit simplement de ne pas tenir compte des connexions des matrices 2 à 4. Les broches MISO des matrices à Leds ne seront pas utilisée et devront restées libres.

Les port P0 à P4 sont utilisés en entrées afin de venir « lire » la position des boutons-poussoirs de direction. A noter que des résistances de tirage de 100 K vers la masse sont à ajouter sur les entrées P0 à P4 (ces résistances ne sont pas représentées sur le schéma ci-dessus – Elles sont en revanche directement présentes sur la platine « CUBLOC Study Board » - Dès lors n'ajoutez pas ces résistances si vous utilisez la platine « CUBLOC Study Board »). Les ports P4 à P7 servent à piloter les entrées de sélection des différentes matrices. Les ports P9 et P10 servent pour la communication SPI™.

Interprétation du programme : (Ce dernier est présent sur notre site Internet)

Le programme ci-dessous concerne la réalisation du jeu du « serpent » en version 4 matrices. Ce dernier est largement commenté et ne devrait pas poser trop de problème de compréhension. Le principe de fonctionnement repose sur l'utilisation d'une variable de type tableau à 2 dimensions qui s'apparente à une « image » des matrices. Ce tableau renferme ainsi la position et l'état de chaque Led des matrices. Le tableau est remis à jour avant chaque rafraîchissement de l'affichage des matrices à Leds. Le début de la partie se matérialise par un « balayage furtif bleu » de l'écran, suivi par l'affichage du serpent en vert (qui ne dispose que d'un seul anneau). La partie ne comence que lorsque le programme détecte que vous avez sollicité le bouton de déplacement vers la droite.

Une seconde variable de type tableau est utilisée pour stocker la position de chaque anneau du serpent (on utilise une variable de type Integer dans laquelle les coordonnées X et Y des anneaux sont placés dans les octets de poids fort et faible). Lorsque le serpent se déplace, on vient lire cette variable pour effacer le dernier anneau qui le constitue.

Une pomme apparaît aléatoirement à l'écran cycliquement à chaque fois que le serpent se déplace de 4 cases. La couleur des pommes ainsi que leur emplacement est aléatoire (bien qu'une fois sur deux on « force » le programme à chosir une pomme rouge : C'est à dire, après 4 déplacement -> pomme rouge, après 4 autres déplacement -> Pomme aléatoire, après 4 autres déplacement -> A nouveau pomme rouge, etc…).

Le fait de privilégier les pommes rouge s'explique que le but principal du jeu est que le serpent soit le plus grand possible.

Le programme accelère également cycliquement la vitesse de déplacement du serpent tout seul afin de rendre la partie plus difficile. Une série de tests permet de détecter quelle pomme a été mangée par le serpent et quelles actions il devra en découler (agrandissement du serpent, accélaration, perte d'un ou plusieurs anneaux, etc…). Le serpent ne doit également pas manger un de ses anneaux (il ne doit donc pas se recouper ou faire demi-tour dès lors qu'il dispose de plus d'un anneau de long). A noter qu'en début de partie, le programme positionne des pommes blanches (mortelles) tout autour de l'air de jeu (dans la partie non visible de l'écran) afin que le jeu s'arrête si le serpent tente de sortir de l'écran.

La version du programme mettant en œuvre 1 seule matrice est très similaire mise à part que les limites de détection de l'aire de jeu seront différentes et qu'il faudra gérer une variable tableau avec moins d'éléments et rafraîchir 1 seule matrice au lieu de 4.

Améliorations possibles :

Nous vous laissons le soin d'apporter des améliorations au programme en ajoutant par exemple des sons, un score, en modifiant la fréquence d'affichage ou le rôle des pommes, etc…

```
'##########################################################################
'# Le jeu du "serpent" sur 4 matrices a leds multi-couleurs Sparfun
'# @Lextronic 2008 - 19/10/2008 - Realise avec CUBLOC Studio Version 2.5.F
##########################################################################

Const Device = CB220

Dim mat1(18,18) As Byte                    ' Memoire RAM matrice 1
Dim memoire(64) As Integer                 ' Memoire deplacement
Dim x As Byte
Dim y As Byte
Dim Direction As Byte                      ' Direction de deplacement
Dim vitesse As Byte                        ' Vitesse deplacement serpent
Dim difficulte As Byte                     ' Niveau du jeu
Dim xserpent As Byte                       ' Derniere position du serpent
Dim yserpent As Byte
Dim taille As Byte                         ' Taille du serpent
Dim Aleatoire1 As Integer                  ' Affichage aleatoire pommes
Dim aleatoire2 As Integer
Dim i As Byte
Dim compt As Byte

High 4                                     ' Port 4 en sortie pilotage CS matrice 1 = 1
High 5                                     ' Port 5 en sortie pilotage CS matrice 2 = 1
High 6                                     ' Port 6 en sortie pilotage CS matrice 3 = 1
High 7                                     ' Port 7 en sortie pilotage CS matrice 4 = 1
```

```
Input 0                                                    ' Port 0 en entree
Input 1                                                    ' Port 1 en entree
Input 2                                                    ' Port 2 en entree
Input 3                                                    ' Port 3 en entree

'-------- Ecran de presentation -----------

For yserpent = 1 To 16                                     ' Efface la matrice
    For xserpent = 1 To 16
        mat1(xserpent,yserpent)=0
    Next
Next
Gosub  affmat1                                             ' Raffraichissement de la matrice
For yserpent = 16 To 1 Step-1                              ' Affiche matrice en bleu
    For xserpent = 1 To 16
        mat1(xserpent,yserpent)=3
   Next
  Gosub  affmat1                                           ' Raffraichissement de la matrice
Next
Delay 200
For yserpent = 1 To 16                                     ' Efface progressivement la matrice
    For xserpent = 1 To 16
        mat1(xserpent,yserpent)=0
    Next
   Gosub  affmat1                                          ' Raffraichissement de la matrice
   Next
  Delay 100

  '---------------------- Initialisation des variables ------------------------

difficulte = 5                                             ' Initialise difficulte du jeu
vitesse = 5                                                ' Initialise vitesse du jeu
taille = 1                                                 ' Initialise taille du serpent
xserpent = 1                                               ' Position initiale du serpent
yserpent = 4
Direction = 1                                              ' Deplacement vers la droite
mat1(xserpent,yserpent) = 2                                ' Memorisation position du serpent
For x=1 To 16                                              ' Delimine le tour de l ecran avec pomme mortelle
(hors vue)
  mat1(x,0)=7
  mat1(x,17)=7
Next
For y=1 To 16                          ' Delimine le tour de l ecran avec pomme mortelle (hors vue)
  mat1(0,y)=7
  mat1(17,y)=7
Next
memoire(1)=&H104                       ' Memoire dernier anneau du serpent
compt = 32                             ' Decompte acceleration
Gosub  affmat1                          ' Raffraichissement de la matrice

'---------------------- Attend debut du jeu ------------------------

Do                                     ' Attend sollicitation touche pour debut jeu
   Aleatoire1 = Rnd(0)
   If In(1) = 1 Then Exit Do
Loop

Do
  Gosub  affmat1                       ' Raffraichissement de la matrice
```

```
'-------------------- Recupere etat des bouton-poussoirs ---------------------

  If In(0) = 1 Then Direction = 2
  If In(1) = 1 Then Direction = 1
  If In(2) = 1 Then Direction = 3
  If In(3) = 1 Then Direction = 4

vitesse  = vitesse-1
if vitesse = 0 Then
    vitesse = difficulte                                    ' initialise variable vitesse
    compt=compt-1                                           ' Acceleration cyclique du jeu apres un certain temps

    Select Case compt
       Case 0
            If difficulte < 3 Then
                difficulte = 3
            Else
                difficulte = difficulte-1
            Endif
            compt = 32
            Gosub pomme1                                    ' Affiche pomme aleatoire sur position aleatoire

        Case 8
            Gosub pomme1                                    ' Affiche pomme raleatoire sur position aleatoire

        Case 12
             i = 1
            Gosub pomme2                                    ' Affiche pomme rouge sur position aleatoire

        Case 16
            Gosub pomme1                                    ' Affiche pomme raleatoire sur position aleatoire

        Case 20
           i = 1
            Gosub pomme2                                    ' Affiche pomme rouge sur position aleatoire

        Case 24
           Gosub pomme1                                     ' Affiche pomme raleatoire sur position aleatoire

        Case 28
             i = 1
            Gosub pomme2                                    ' Affiche pomme rouge sur position aleatoire

End Select

'-------------------- Mise a jour position du serpent ---------------------

 Select Case Direction
            Case 1                                  ' On deplace le serpent vers la droite
              xserpent = xserpent + 1
            Case 2                                  ' On deplace le serpent vers la gauche
              xserpent = xserpent -1
            Case 3                                  ' On deplace le serpent vers le haut
              yserpent = yserpent -1
            Case 4                                  ' On deplace le serpent vers le bas
              yserpent = yserpent +1
End Select
```

```
i = mat1(xserpent,yserpent)          ' Recupere ce qu'il y a a la nouvelle position du serpent

  Select Case i
    Case 0                           ' Aucun obstacle, le serpent se deplace normalement
        Gosub avance

    Case 1                           ' Cas d'une pomme rouge
        mat1(xserpent,yserpent) = 2                                  ' Memorisation position du serpent
        taille = taille + 1                                          ' Le serpent grandi
        If taille > 63 Then taille = 63                              ' test si taille maxi
          memoire(taille).Byte1 = xserpent
          memoire(taille).Byte0 = yserpent

    Case 2                                      ' Le serpent se recoupe ... Fin de partie
        Gosub finpart
        Reset                                   ' La partie recommence

    Case 3                           ' Cas d'une pomme Bleu fonce... Le serpent perd un anneau
        Gosub avance
        Gosub reduit                 ' Enleve 1 anneau

    Case 4                           ' Pomme orange continu a avancer... mais accelere le jeu
        Gosub avance
        If difficulte <> 1 Then
          difficulte = difficulte-1
        Endif
    Case 5                           ' Cas d'une pomme Bleu clair... Le serpent perd 3 anneaux
        Gosub avance
        Gosub reduit                 ' Enleve 1 anneau
        Gosub reduit                 ' Enleve 1 anneau
        Gosub reduit                 ' Enleve 1 anneau

    Case 6                           ' Pomme violette continu a avancer... mais reduit vitesse du jeu
        Gosub avance
        difficulte = difficulte + 1

    Case 7                           ' Le serpent a touche une pomme mortelle ... Fin de partie
        Gosub finpart
        Reset                        ' La partie recommence

 End Select

  Endif

Loop

          '---------------------- Rafraichissement de la matrice -----------------------
affmat1:
          Low 4
                ' Selection CS matrice 1
          For y = 1 To 8
            For x = 1 To 8
              Shiftout 9,10,1,mat1(x,y),8
            Next
    Next
          High 4
' Desactivation CS matrice 1
```

```
            Low 5
                    ' Selection CS matrice 2
            For y = 1 To 8
              For x = 9 To 16
                Shiftout 9,10,1,mat1(x,y),8
              Next
    Next
            High 5
          ' Desactivation CS matrice 2

            Low 6
                    ' Selection CS matrice 3
            For y = 9 To 16
              For x = 1 To 8
                Shiftout 9,10,1,mat1(x,y),8
              Next
    Next
            High 6
          ' Desactivation CS matrice 3

                    Low 7
                            ' Selection CS matrice 4
            For y = 9 To 16
              For x = 9 To 16
                Shiftout 9,10,1,mat1(x,y),8
              Next
    Next
            High 7
          ' Desactivation CS matrice 4
                    Return

            '---------------------- Animation couleur du serpent en fin de partie ----------------------
finpart:
                    For i = 1 To 5
                      For yserpent = 1 To 16                                    ' Le serpent change de couleur
plusieurs fois
                  For xserpent = 1 To 16
                    If mat1(xserpent,yserpent)=2 Then mat1(xserpent,yserpent)=7
                  Next
                              Next
               Gosub  affmat1
          ' Raffraichissement de la matrice
                    For yserpent = 1 To 16                                      ' Efface
progressivement la matrice
                  For xserpent = 1 To 16
                    If mat1(xserpent,yserpent)=7 Then mat1(xserpent,yserpent)=4
                  Next
                       Next
              Gosub  affmat1
          ' Raffraichissement de la matrice
                              For yserpent = 1 To 16                            ' Efface
progressivement la matrice
                  For xserpent = 1 To 16
                    If mat1(xserpent,yserpent)=4 Then mat1(xserpent,yserpent)=2
                  Next
                              Next
              Gosub  affmat1
          ' Raffraichissement de la matrice
                    Next
    Return

            '---------------------- Le serpent avance ----------------------
avance:
```

```
    mat1(xserpent,yserpent) = 2                    ' Memorisation position du serpent
    x = memoire(1).Byte1
    y = memoire(1).Byte0
    mat1(x,y) = 0                                  ' Efface le dernier anneau du serpent sur l'ecran
    For x = 1 To 63                                ' Decalle position memoire
       memoire(x)=memoire(x+1)
    Next
    memoire(taille).Byte1 = xserpent
    memoire(taille).Byte0 = yserpent
Return

'---------------------- Le serpent perd  un anneau -----------------------
reduit:
   If taille > 1 Then
      taille = taille -1
      x = memoire(1).Byte1
      y = memoire(1).Byte0
      mat1(x,y) = 0                                ' Efface le dernier anneau du serpent sur l'ecran
      For x = 1 To 63                              ' Decalle position memoire
        memoire(x)=memoire(x+1)
      Next
   Endif
Return

'---------------------- Affiche pommes alatoirement -----------------------

pomme1:
  Aleatoire1 = Rnd(0)            ' Generation pomme aleatoire
  i = Aleatoire1/4285
  If i=0 Then i = 1
  If i=2 Then i = 1
  If i>7 Then i = 1

pomme2:
  aleatoire2 = Rnd(0)
  x = aleatoire2.Byte1                             ' Generation coordonnees aleatoire de la pomme
  y = aleatoire2.Byte0
  x = x / 9                                        ' Repositionne x entre 0 et 16
  y = y / 9                                        ' Repositionne y entre 0 et 16
  If x = 0 Then x = 1
  If y = 0 Then y = 1
  If x > 16 Then x = 16
  If y > 16 Then y = 16
  If mat1(x,y) = 0 Then mat1(x,y)=i                ' Regarde si il n y a pas deja une pomme ou le serpent
Return
```

NOTE D'APPLICATION # 47.
Pilotez un module de suivi de ligne

Cette note d'application va vous permettre de piloter le module infrarouge « I2CLF » au moyen d'un CUBLOC CB220B. Ce module électronique conçu par la société Robotics Connexion permettra à votre robot de suivre une ligne au sol au moyen de 5 capteurs infrarouges. Ces capteurs sont capables de faire la distinction entre les couleurs blanches et noires afin de pouvoir suivre une ligne noire sur un sol blanc (ou inversement).

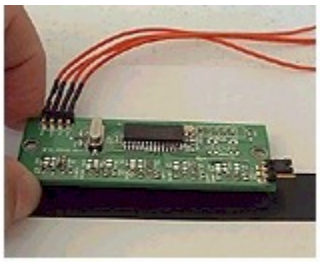
D'usage universel, ce module de faibles dimensions (75 x 27 mm) pourra être interfacé avec tout type de microcontrôleur disposant d'un bus I2C™ (comme un CB220B par exemple). Sur simple interrogation, la platine retourne un octet dont les bits indiquent ce que les capteurs détectent.

Notions abordées :
- Communication I2C

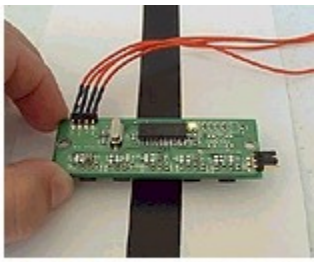
Matériel nécessaire :
- Un module « CB220B »
- Un module « I2CLF »

Description de l'application:
Cette application vous permettra depuis l'écran de Débug du Cubloc Studio d'afficher en temps réel l'état des 5 capteurs du module « I2CLF ».

L'écran de débug retourne
[x] [x] [x] [x] [x]



L'écran de débug retourne
[ ] [ ] [ ] [ ] [ ]

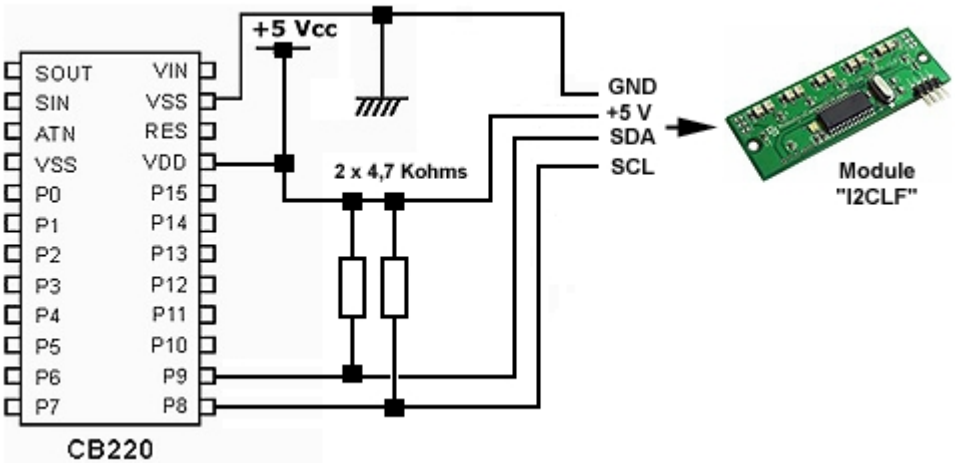


L'écran de débug retourne
[x] [x] [ ] [x] [x]

L'état de chaque Led infrarouge sera ainsi affiché entre « croché ». Un « X » signifiera que la led infrarouge détecte un fond clair tandis qu'une absence de « X » signifiera qu'elle détecte une couleur foncée. Attention la ligne noire devra être réalisée dans un matériau réflechissant et elle devra être d'une largeur minimale de 4 cm.

Préparation matérielle:

Cette application nécessite que vous réalisiez le schéma ci-dessous.



Ce dernier est très simple et nécessite simplement de raccorder les broches 8 et 9 du module CUBLOC CB220B (lesquelles seront reconfigurées pour une liaison I2C™) aux broches SDA et SCL du module infrarouge « I2CLF ».

Interprétation du programme : (Ce dernier est présent sur notre site Internet)

Le programme ci-dessous (qui nous a été généreusement envoyé par Garret Mace de la division CUBLOC USA) est très simple à interpréter.

Ce dernier commence par la déclaration du port I2C™ sur le CB220 (lequel utilisera les broches 8 et 9). S'en suit la lecture des données du module « I2CLF », lequelles seront stockées dans la variable I. Le programme va ensuite tester un à un l'état des 5 premiers bits de la variable I (lesquels reflètent l'état de détection des 5 Leds infrarouges du déteteur). En fonction des valeurs lues, le programme affichera ou non la lettre X entre les crochés.

```
'##########################################################################
'# Pilotage d'un module de détection de ligne à commande I2C™
'# @Garret Mace - 09/04/2010
##########################################################################

Const Device = CB220
Ramclear

Dim i As Byte

Set I2c 9,8

Do

  I2cstart       ' Lecture des données du module I2CLF
  i = I2cwrite(0x51)
```

```
  i = I2cread(0)
  I2cstop

Debug Goxy,0,0

If i.bit0 = 0 Then    ' Test l'état de la led infrarouge 1
   Debug "[X] "
Else
   Debug "[ ] "
Endif

If i.bit1 = 0 Then    ' Test l'état de la led infrarouge 2
   Debug "[X] "
Else
   Debug "[ ] "
Endif

If i.bit2 = 0 Then    ' Test l'état de la led infrarouge 3
   Debug "[X] "
Else
   Debug "[ ] "
Endif

If i.bit3 = 0 Then    ' Test l'état de la led infrarouge 4
   Debug "[X] "
Else
   Debug "[ ] "
Endif

If i.bit4 = 0 Then    ' Test l'état de la led infrarouge 5
   Debug "[X]"
Else
   Debug "[ ]"
Endif

Delay 50

Loop
```