Edité le 3 novembre 2005

PROTOCOLE Extralink

Principe de dialogue : maitre esclave.

Le caractère 27 (ESC) est le caractère de début du message émis par l'ordinateur.
Le caractère 28 (FS) est le caractère de séparation des sous-messages
Le caractère 29 (GS) est le caractère de fin du message émis par l''ordinateur.
Le caractère 30 est le caractère de fin de la réponse émise par le pack Extralink.

Ces quatre caractères ainsi que le caractère 26 (SUB) sont réservés. Si l'un de ces caractères doit être émis dans le corps du message, il sera remplacé par le caractère 26 (SUB) suivi d'un caractère de la manière suivante :

<26> sera remplacé par <26> <0>
<27> sera remplacé par <26> <1>
<28> sera remplacé par <26> <2>
<29> sera remplacé par <26> <3>
<30> sera remplacé par <26> <4>

Un message émis par l'ordinateur est toujours terminée par la checksum, qui est la somme sur 8 bits de tous les octets précédents, caractère ESC de début de message inclus, et suivi du caractère 29.

Dans le cas d’un module à adresse figée, la réponse envoyée par le pack est terminée par un octet qui indique le status de la carte de base, suivi des informations, de la checksum et du caractère 30.

Dans le cas d’un module réadressable, la réponse envoyée par le pack est terminée par un octet qui indique le status de la carte de base, suivi d’un octet qui indique le status du module, et suivi des informations, de la checksum et du caractère 30.

La checksum est égale à 255 moins la somme sur 8 bits des octets précédents.

Les types de message

Il existe 3 types de message :

1 - message d'écriture (module à adresse figée)

L'ordinateur envoie :

ESC - ADRSLV – LNG - W1 - W2 ... Wn - CKW - 28

Où :
ESC = 27
ADRSLV = adresse du module esclave concerné multipliée par 2
LNG = longueur de la réponse retournée par le module
W1 ... Wn = octets envoyés au module
CKW = checksum (somme sur 8 bits de tous les octets précédents, ESC inclus)

Le pack répond :
STATUS - CKR - 30

Où :
STATUS = rapport de transmission
CKR = checksum (255 - somme sur 8 bits des octets précédents)

2 - message de lecture (module à adresse figée)

ESC - ADRSLV + 1 – LNG - CKW - 29

Où :
ESC = 27
ADRSLV = adresse du module esclave concerné multipliée par 2
LNG = longueur de la réponse retournée par le module
CKW = checksum (somme sur 8 bits de tous les octets précédents , ESC inclus)

L'unité répond :
STATUS - R1- R2 - ... Rn - CKR - 30

Où :
STATUS = rapport de transmission
R1 ... Rn = octets envoyés par le module
CKS = checksum (255 - somme sur 8 bits de tous les octets précédents)

3 - message d'écriture - lecture (modules réadressables)

ESC - ADRSLV + 1 – LNG - W1 - W2 ... Wn - CKW - 29

Où :
ESC = 27
ADRSLV = adresse du module esclave concerné * 2
LNG = longueur de la réponse retournée par le module
W1 ... Wn = octets envoyés au module
CKM = checksum (somme sur 8 bits de tous les octets précédents (ESC inclus)

Le pack répond :
STATUS – STMOD - R1- R2 - ... Rn - CKR - 30

Où :
STATUS = rapport de transmission
STMOD = status module réadressable
R1 ... Rn = octets envoyés par le module
CKS = checksum (255 - somme sur 8 bits de tous les octets précédents

Les bases se comportent comme un module réadressable situé à l’adresse 8. Dans ce cas, STMOD = 0.

Programmation

Le programme qui gère les communication peut etre lu sous éditeur de textes dans le fichier « XA_DLL_reduit.pas ». Ce fichier est extrait de « XA_DLL.pas» qui est le source de la DLL Extraklink, dont on a expurgé ce qui concerne les types 0 (USB), 4 (Ethernet) et 3 (radio).

Commentaires :

Les fonctions série sont les fonctions de base d'émission-reception sur un port COM au travers d'un DCB Windows.

XA_open et XA-close se contentent d'ouvrir et de fermer le port après vérification.

XA_insere_car_recu insère un caractère dans le buffer de reception (Rec_buffer) sauf si ce caractère est un séparateur (28) en quel cas il met à jour la table des séparateurs TabSS. Ce dernier pont peut être omis si on n'utilise pas les séparateurs.

XA_raw_insertB et XA-insertB insèrent un octet dans le buffer du message émis (out_buffer) avec ou avec trancodification.

XA_createmsg initialise le buffer d'émission et place 27 en tête s'il s'agit d'un nouveau message, ou place le séparateur 28 sinon. Ce dernier pont peut être omis si on n'utilise pas les séparateurs.

XA_build complète le message en y placant : l'adresse du module, la longueur de la réponse attendue et le code fonction.

XA_send termine d'abord le message en calculant la checksum, et ajoute le caactère de fin 29. Il effectue ensuite l'émission-reception du message, vérifie le checksum et initialise les variables pour lecture par les fonctions XA_get qui suivent.

Les fonctions **XAN et XAN_str** sont équivalentes à XA et XA_str avec un code fonction numérique plutôt qu'une chaine. Elles construisent et envoient le message selon la structure donnée par le code fonction.

Tables des fonctions

Les fonctions Fxx correspondent à la valeur numérique xx (F5 correspond à 5).

20 = 'WRB'
21 = 'SETBIT'
21 = 'WRB1'
22 = 'RSTBIT'
22 = 'WRB2'
30 = 'WRI'
31 = 'WRI1'
32 = 'WRI2'
40 = 'WRL'
41 = 'WRL1'
42 = 'WRL2'
43 = 'WRL3'
50 = 'HITRANS'
50 = 'WRCB'
51 = 'LOTRANS'
51 = 'WRCB1'
52 = 'HILOTRANS'
52 = 'WRCB2'
53 = 'WRCB3'
54 = 'WRCB4'
55 = 'WRCB5'
56 = 'WRCB6'
60 = 'DELAY'
60 = 'WRCI'
61 = 'BLINK'
61 = 'WRCI1'
62 = 'WRCI2'
80 = 'LCDSET'
81 = 'LCDSTR'
81 = 'WRSTR'
82 = 'LCDSTRPOS'
98 = 'WRMEM'
99 = 'RDMEM'
100 = 'RDB'
101 = 'RDI'
101 = 'RDW'
102 = 'RDBI'
103 = 'RDL'
104 = 'RDBL'
105 = 'RDIL'
106 = 'RDBIL'
107 = 'RDLL'
108 = 'RDBLL'
109 = 'RDWLL'
111 = 'RDLLL'
120 = 'RDCB'
121 = 'RDCW'
121 = 'RDCI'
123 = 'RDCL'
127 = 'RDCLL'
140 = 'SET629'
200 = 'OPEN'
201 = 'CLOSE'
202 = 'ADRPACK'
202 = 'ADRIP'
203 = 'SETADR'
204 = 'LEDON'
205 = 'LEDOFF'
206 = 'RDVER'
207 = 'RSTMOD'
208 = 'TIMOUT'
209 = 'SUBMSG'
210 = 'I2CWR1'
211 = 'I2CWR2'
212 = 'I2CWR3'
213 = 'I2CWR4'
214 = 'I2CWR5'
215 = 'I2CWR6'
220 = 'I2CRD'
250 = 'TJ'
251 = 'TK'

Table des messages

0	'OK'
1	'Base not initializated'
11	'Base frame error'
12	'Base Overrun error'
13	'Base I2C time out'
14	'Base too long message'
15	'Base Escape character error'
16	'Base checksum error'
17	'Base message length error'
18	'Base received message too short'
19	'Module unknown function'
101	'Not initialized'
102	'Open error'
103	'Comm not open'
104	'Message empty'
105	'No answer or truncated answer'
106	'Erreur checksum from pack'
107	'Too long answer'
108	'Bad argument '
109	'Bad character inside answer'
110	'IP send error'
111	'IP receive error'
112	'IP Close error'
113	'Other send error'
114	'IP connexion error'
115	'Too many submessages'
116	'Too long message'
240	'Unknown command from module'
255	'Unknown function string'