ELECTRONIQUE DE COMMANDE POUR MAIN ROBOTISEE

PROTOCOLE DE COMMUNICATION

1. Introduction

Chaque moteur est mécaniquement couplé à un capteur de position de type codeur incrémental. L'ensemble ainsi formé est appelé voie. Les voies sont numérotées de 0 à 5 comme suit :

- Voie 0 = Rotation Pouce
- Voie 1 = Pouce
- Voie 2 = Index
- Voie 3 = Majeur
- Voie 4 = Annulaire
- Voie 5 = Auriculaire

A chaque voie est affecté un ensemble de registres définissant son fonctionnement. Ces registres sont regroupés dans un tableau (1 par voie) dans un ordre défini. Le N° d'ordre dans le tableau d'un registre donné est appelé N° de registre. (La numérotation des registres commence à 0)

L'accès à un registre particulier d'une voie donnée se fait par l'intermédiaire de sa position mémoire. La position mémoire est calculée de la manière suivante :

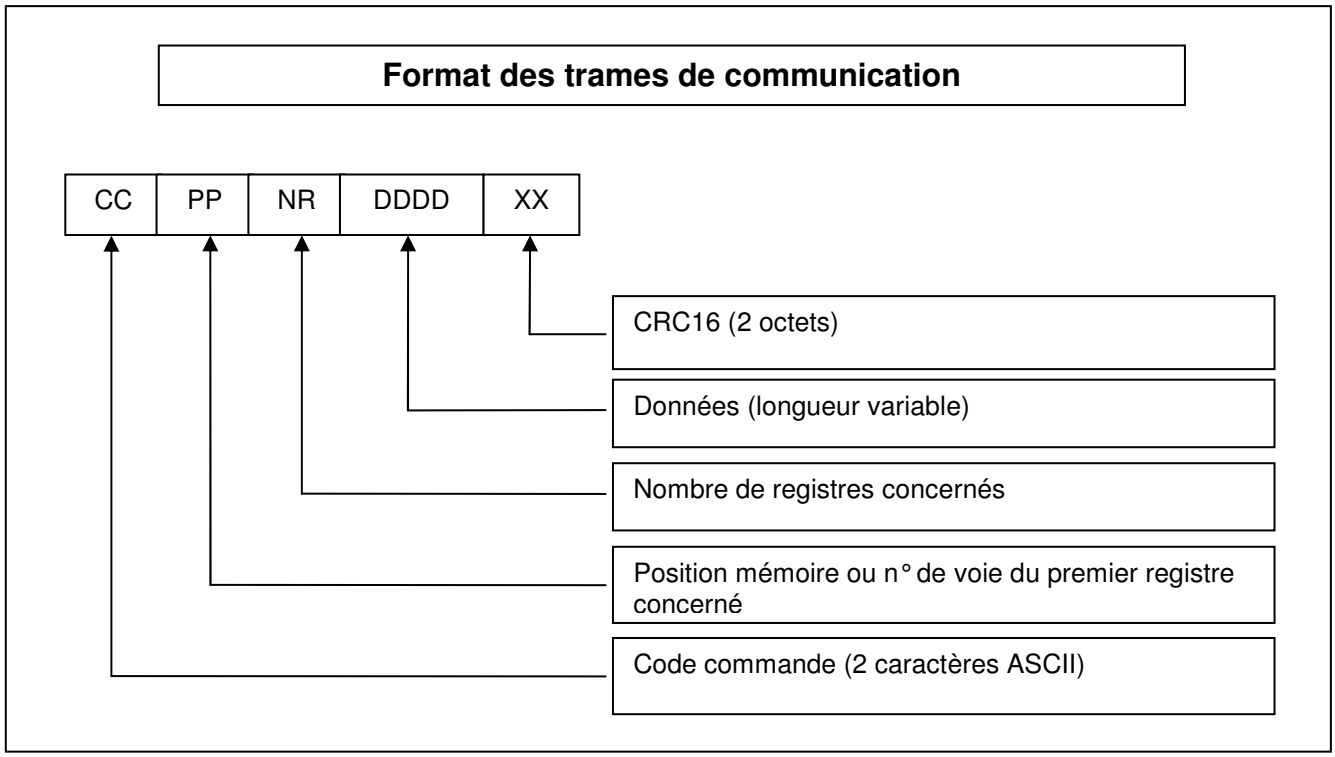
- Position mémoire = ((N° de Voie + 1) * 1000) + N° du registre.

- Exemple : La position mémoire du registre N°5 de la voie 2 est : ((2 + 1) * 1000) + 5 = 3005

2. Généralités

La main robotisée intègre un composant qui réalise l'interface entre un port USB du PC et un port UART du microcontrôleur de sorte que la liaison soit vue de part et d'autre comme une liaison série de type port COM. La vitesse de communication est de **460 800** bauds.

La communication entre le PC et la main est basée sur des lectures et écritures de registres. Elle est toujours initiée par le PC qui envoie une trame de requête à laquelle la main répond par une trame de réponse.



Les trames commencent toujours par un code de commande servant à identifier le type de requête.

Les codes de commandes implémentés à ce jour sont :

- RD : Lecture de "n" registres à partir d'une position mémoire donnée

- WR : Ecriture de"n" registres à partir d'une position mémoire donnée

- W1 : Ecriture des registres "MODE_CMD_MOTEUR" de "n" voies consécutives et lecture des registres "POSITION_CODEUR" des "n" voies.

- W2 : Ecriture des registres "CONSIGNE_TENSION_POSITION" de "n" voies consécutives et lecture des registres "POSITION_CODEUR" des "n" voies.

- W3 : Ecriture des registres "LIMITE_COURANT" de "n" voies consécutives et lecture des registres "POSITION_CODEUR" des "n" voies.

- W4 : Ecriture des registres "MODE_CMD_MOTEUR" de "n" voies consécutives et lecture des registres "VITESSE_MOTEUR" des "n" voies.

- W5 : Ecriture des registres "CONSIGNE_TENSION_POSITION" de "n" voies consécutives et lecture des registres "VITESSE_MOTEUR" des "n" voies.

- W6 : Ecriture des registres "LIMITE_COURANT" de "n" voies consécutives et lecture des registres "VITESSE_MOTEUR" des "n" voies.

- BL : Demande de chargement d'un nouveau programme dans le microcontrôleur.

Les commande RD et WR permettent de lire ou d'écrire une suite de registres en indiquant la position mémoire du premier registre à lire ou écrire.

Les commandes W1 à W6 effectuent des lectures et écritures de registres en un seul échange. D'une manière générale, elles permettent d'écrire l'un des registres de n voies consécutives et d'obtenir en réponse l'un des registres de ces n voies.

Toutes les valeurs numériques sont transmises directement en hexadécimal en commençant par l'octet de poids le plus faible et en allant jusqu'à l'octet de poids le plus fort.

Les transmissions sont sécurisées par l'ajout d'un CRC16 en fin de trame. (Voir exemples de programme de calcul en fin de document)

A réception d'une trame, le microcontrôleur vérifie que :

- Le CRC16 reçu est identique à celui qu'il a recalculé à partir des octets reçus.
- Le code commande est connu.
- Les positions de départ et de fin sont contenues dans sa zone mémoire
- La commande demandée est autorisée dans la zone mémoire spécifiée. (Pas d'écriture des registres en lecture seule)

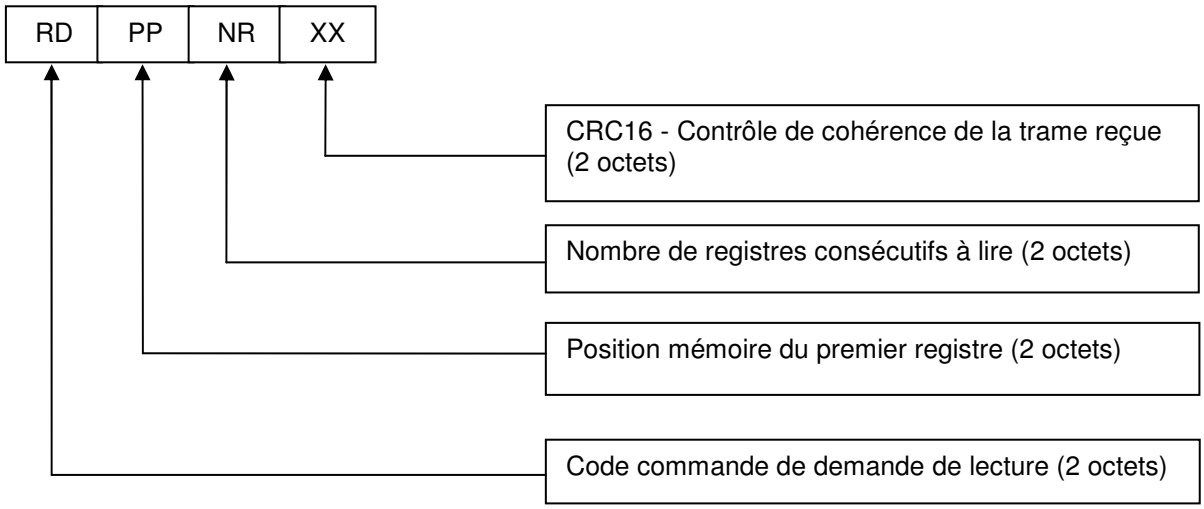
Si toutes ces conditions sont remplies, la trame est prise en compte et une réponse est retournée.

Le microcontrôleur détecte la fin d'une trame lorsqu'il ne reçoit plus de donnée pendant 100us

Lorsque aucune trame n'est reçue du PC pendant 500ms, les voies configurées pour fonctionner en mode consigne de tension sont automatiquement désactivées.
Les registres (MODE_CMD_MOTEUR) et (CONSIGNE_TENSION_POSITION) remis à 0.

Code commande RD - Format de la trame émise par le PC

Demande de lecture de n registres consécutifs à partir d'une position mémoire donnée



Exemple : Demande de lecture de 2 registres consécutifs à partir de la position 1000 :

Octets émis : 0x52, 0x44, 0xE8, 0x03, 0x02, 0x00, 0x39, 0x66

0x52 = Code ascii du caractère R
0x44 = Code ascii du caractère D

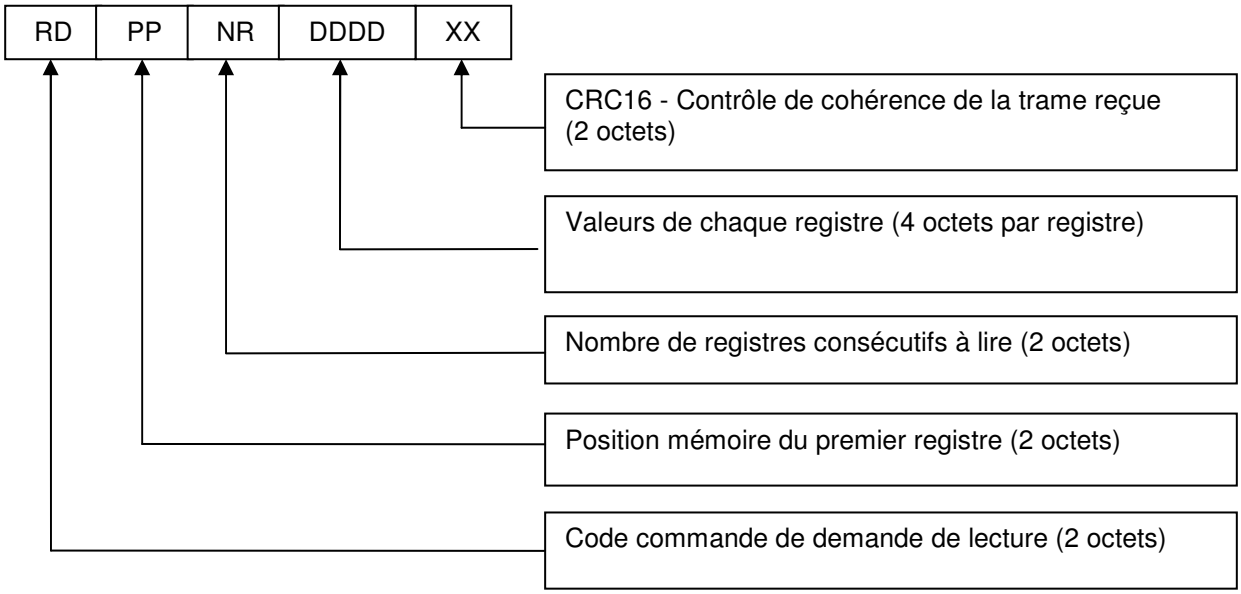
0xE8 = Poids faible la première position mémoire
0x03 = Poids fort de la première position mémoire

0x02 = Poids faible du nombre de registre
0x00 = Poids fort du nombre de registre

0x39 = Poids faible du CRC16
0x66 = Poids fort du CRC16

Code commande RD - Format de la trame renvoyée par la main

Réponse à une demande de lecture de n registres consécutifs à partir d'une position mémoire donnée



Exemple : Réponse à une demande de lecture de 2 registres consécutifs à partir de la position 1000

Octets reçus : 0x52, 0x44, 0xE8, 0x03, 0x02, 0x00, 0x01, 0x00, 0x00, 0x00, 0xA8, 0x61, 0x00, 0x00, 0x75, 0x0A

0x52 = Code ascii du caractère R
0x44 = Code ascii du caractère D

0xE8 = Poids faible de la première position mémoire
0x03 = Poids fort de la première position mémoire

0x02 = Poids faible du nombre de registre
0x 00 = Poids fort du nombre de registre

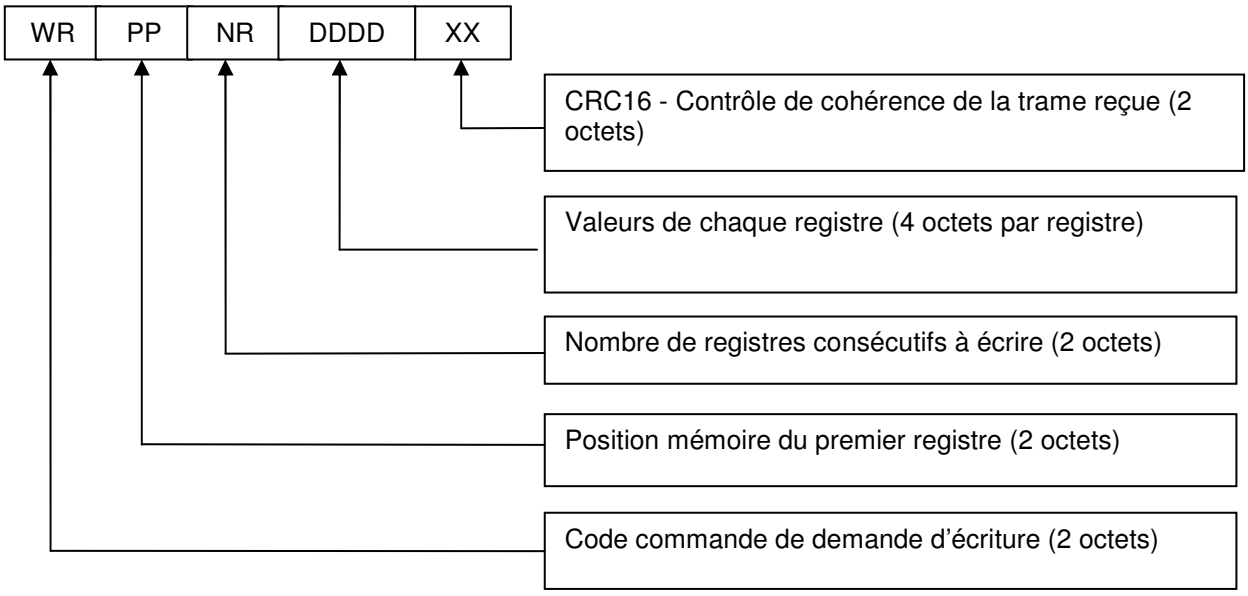
0x01 = Poids le plus faible de la valeur du registre situé à l'adresse 1000
0x00
0x00
0x00 = Poids le plus fort de la valeur du registre situé à l'adresse 1000

0xA8 = Poids le plus faible de la valeur du registre situé à l'adresse 1001
0x61
0x00
0x00 = Poids le plus fort de la valeur du registre situé à l'adresse 1001

0x75 = Poids faible du CRC16
0x0A = Poids fort du CRC16

Code commande WR - Format de la trame émise par le PC

Demande d'écriture de n registres consécutifs à partir d'une position mémoire donnée



Exemple : Demande d'écriture de 2 registres consécutifs à partir de la position 1000
(1 à la position 1000 et 25000 à la position 1001)

Octets émis : 0x57, 0x52, 0xE8, 0x03, 0x02, 0x00, 0x01, 0x00, 0x00, 0x00, 0xA8, 0x61, 0x00, 0x00, 0x47, 0x59

0x57 = Code ascii du caractère W
0x52 = Code ascii du caractère R

0xE8 = Poids faible de la première position mémoire
0x03 = Poids fort de la première position mémoire

0x02 = Poids faible du nombre de registre
0x00 = Poids fort du nombre de registre

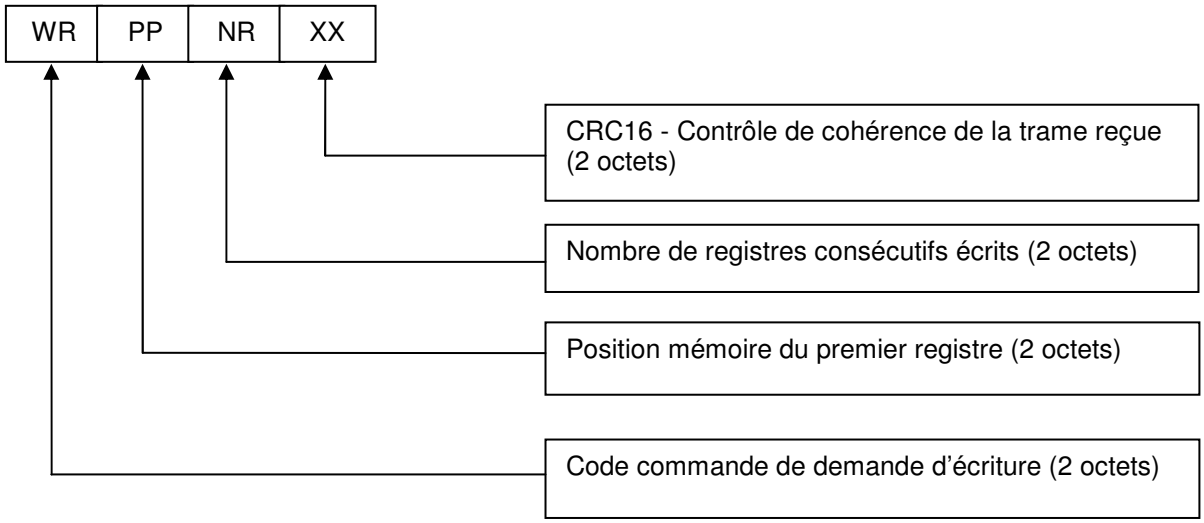
0x01 = Poids le plus faible de la valeur à écrire dans le registre à la position 1000
0x00
0x00
0x00 = Poids le plus fort de la valeur à écrire dans le registre à la position 1000

0xA8 = Poids le plus faible de la valeur à écrire dans le registre à la position 1001
0x61
0x00
0x00 = Poids le plus fort de la valeur à écrire dans le registre à la position 1001

0x47 = Poids faible du CRC16
0x59 = Poids fort du CRC16

Code commande WR - Format de la trame renvoyée par la main

Réponse à une demande d'écriture de n registres consécutifs à partir d'une position mémoire donnée



Exemple : Réponse à une demande d'écriture de 2 registres consécutifs à partir de la position 1000

Octets reçus : 0x57, 0x52, 0xE8, 0x03, 0x02, 0x00, 0x70, 0xF0

0x57 = Code ascii du caractère W
0x52 = Code ascii du caractère R

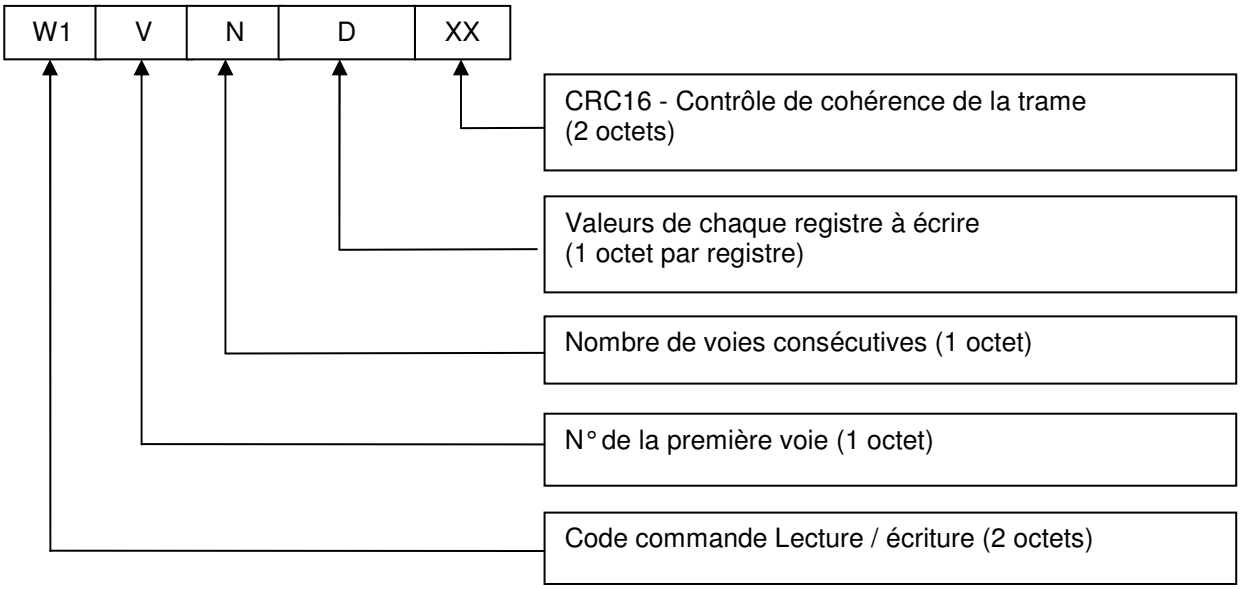
0xE8 = Poids faible la première position mémoire
0x03 = Poids fort de la première position mémoire

0x02 = Poids faible du nombre de registre
0x00 = Poids fort du nombre de registre

0x70 = Poids faible du CRC16
0xF0 = Poids fort du CRC16

Code commande W1 - Format de la trame émise par le PC

Demande d'écriture du registre "MODE_CMD_MOTEUR" de n voies consécutives et
Lecture du registre "POSITION_CODEUR" des n voies



Exemple : Demande d'écriture des registres "MODE_CMD_MOTEUR" des voies 1 à 3
Et Lecture des registres " POSITION_CODEUR" des voies 1 à 3

Octets émis 0x57, 0x31, 0x01, 0x03, 0x02, 0x01, 0x00, 0xA4, 0x30

0x57 = Code ascii hexadécimal du caractère W
0x31 = Code ascii hexadécimal du caractère 1

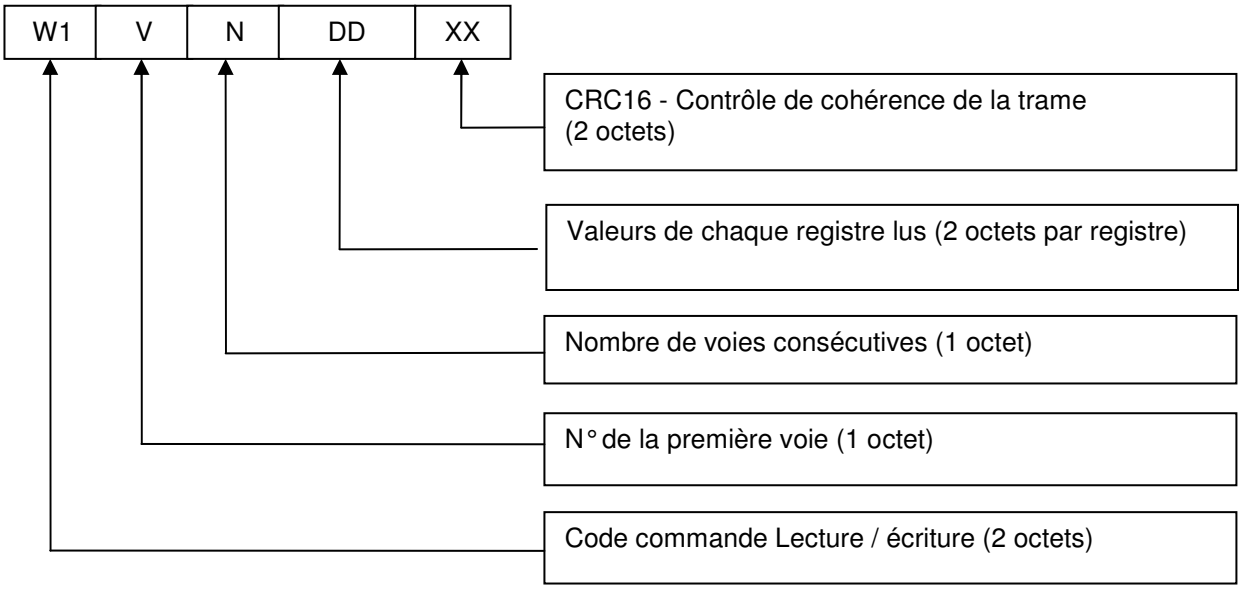
0x01 = N° de la première voie
0x03 = Nombre de voies

0x02 = Valeur à écrire dans le registre "MODE_CMD_MOTEUR" de la voie 1
0x01 = Valeur à écrire dans le registre "MODE_CMD_MOTEUR" de la voie 2
0x00 = Valeur à écrire dans le registre "MODE_CMD_MOTEUR" de la voie 3

0xA4 = Poids faible du CRC16
0x30 = Poids fort du CRC16

Code commande W1 - Format de la trame renvoyée par la main

Réponse à une demande d'écriture du registre "MODE_CMD_MOTEUR" de n voies consécutives et Lecture du registre "POSITION_CODEUR" des n voies



Exemple : Réponse à une demande d'écriture des registres "MODE_CMD_MOTEUR" des voies 1 à 3
Et Lecture des registres "POSITION_CODEUR" des voies 1 à 3

Octets émis 0x57, 0x31, 0x01, 0x03, 0xE8, 0x03, 0x00, 0x00, 0x20, 0x4E, 0xA3, 0x1D

0x57 = Code ascii du caractère W
0x31 = Code ascii du caractère 1

0x01 = N° de la première voie
0x03 = Nombre de voies

0xE8 = Poids faible du registre "POSITION_CODEUR" de la voie 1
0x03 = Poids fort du registre "POSITION_CODEUR" de la voie 1

0x00 = Poids faible du registre "POSITION_CODEUR" de la voie 2
0x00 = Poids fort du registre "POSITION_CODEUR" de la voie 2

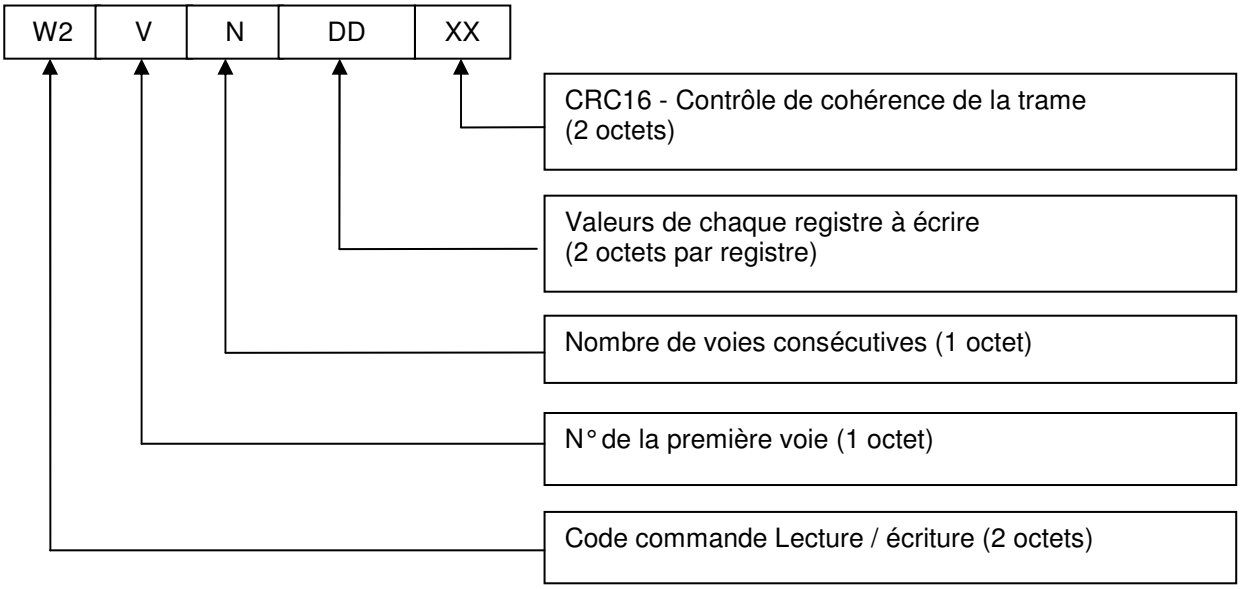
0x20 = Poids faible du registre "POSITION_CODEUR" de la voie 3
0x4E = Poids fort du registre "POSITION_CODEUR" de la voie 3

0xA3 = Poids faible du CRC16
0x1D = Poids fort du CRC16

Attention : Pour ce type commande, la valeur de "POSITION_CODEUR" est retournée sur 2 octets et doit être considérée comme positive lorsqu'elle est inférieure à 60535 et négative lorsqu'elle est supérieure à cette valeur

Code commande W2 - Format de la trame émise par le PC

Demande d'écriture du registre "CONSIGNE_TENSION_POSITION" de n voies consécutives et
Lecture du registre "POSITION_CODEUR" des n voies



Exemple : Demande d'écriture des registres "CONSIGNE_TENSION_POSITION" des voies 1 à 3
Et Lecture des registres " POSITION_CODEUR" des voies 1 à 3

Octets émis 0x57, 0x32, 0x01, 0x03, 0x20, 0x03, 0x00, 0x00, 0xF4, 0x01, 0xB9, 0x51

0x57 = Code ascii hexadécimal du caractère W
0x32 = Code ascii hexadécimal du caractère 2

0x01 = N° de la première voie
0x03 = Nombre de voies

0x20 = Poids faible de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 1
0x03 = Poids fort de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 1

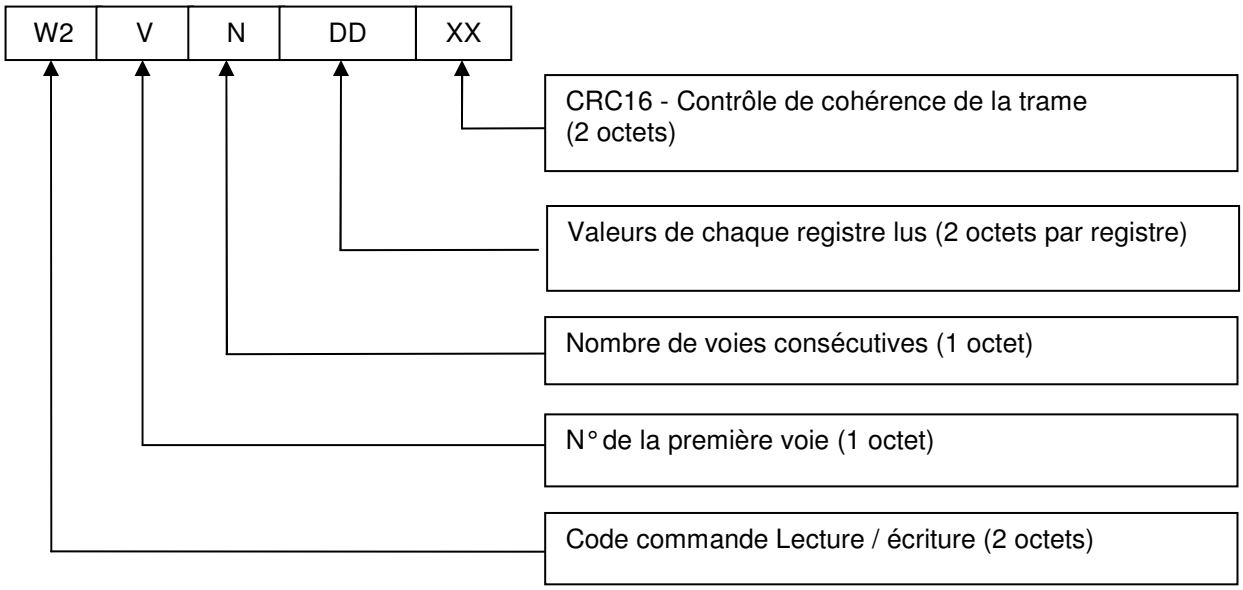
0x00 = Poids faible de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 2
0x00 = Poids fort de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 2

0xF4 = Poids faible de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 3
0x01 = Poids fort de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 3

0xB9 = Poids faible du CRC16
0x51 = Poids fort du CRC16

Code commande W2 - Format de la trame renvoyée par la main

Réponse à une demande d'écriture du registre "CONSIGNE_TENSION_POSITION" de n voies consécutives et lecture du registre "POSITION_CODEUR" des n voies



Exemple : Réponse à une demande d'écriture des registres " CONSIGNE_TENSION_POSITION " des voies 1 à 3 ett Lecture des registres "POSITION_CODEUR" des voies 1 à 3

Octets émis 0x57, 0x32, 0x01, 0x03, 0xE8, 0x03, 0x00, 0x00, 0x20, 0x4E, 0xB7, 0xED,

0x57 = Code ascii du caractère W
0x32 = Code ascii du caractère 2

0x01 = N° de la première voie
0x03 = Nombre de voies

0xE8 = Poids faible du registre "POSITION_CODEUR" de la voie 1
0x03 = Poids fort du registre "POSITION_CODEUR" de la voie 1

0x00 = Poids faible du registre "POSITION_CODEUR" de la voie 2
0x00 = Poids fort du registre "POSITION_CODEUR" de la voie 2

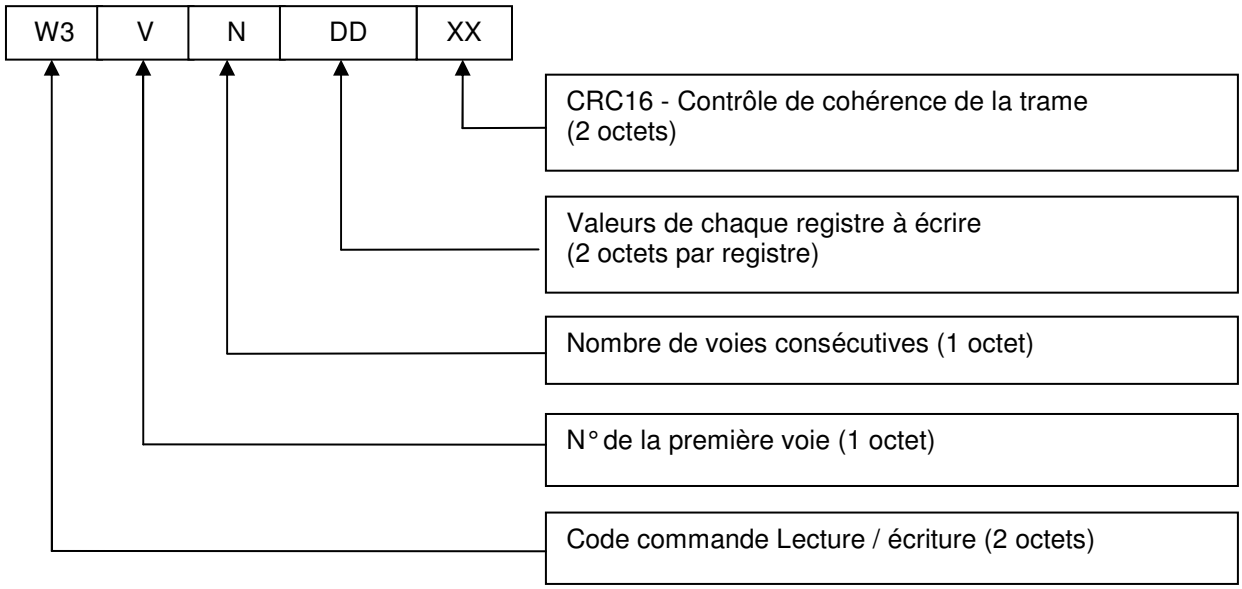
0x20 = Poids faible du registre "POSITION_CODEUR" de la voie 3
0x4E = Poids fort du registre "POSITION_CODEUR" de la voie 3

0xB7 = Poids faible du CRC16
0xED = Poids fort du CRC16

Attention : Pour ce type commande, la valeur de "POSITION_CODEUR" est retournée sur 2 octets et doit être considérée comme positive lorsqu'elle est inférieure à 60535 et négative lorsqu'elle est supérieure à cette valeur

Code commande W3 - Format de la trame émise par le PC

Demande d'écriture du registre "LIMITE_COURANT" de n voies consécutives et
Lecture du registre "POSITION_CODEUR" des n voies



Exemple : Demande d'écriture des registres " LIMITE_COURANT" des voies 1 à 3
Et Lecture des registres " POSITION_CODEUR" des voies 1 à 3

Octets émis 0x57, 0x33, 0x01, 0x03, 0x30, 0x75, 0x00, 0x00, 0x20, 0x4E, 0x61, 0x6E

0x57 = Code ascii du caractère W
0x33 = Code ascii du caractère 3

0x01 = N° de la première voie
0x03 = Nombre de voies

0x30 = Poids faible de la valeur à écrire dans le registre "LIMITE_COURANT" de la voie 1
0x75 = Poids fort de la valeur à écrire dans le registre "LIMITE_COURANT " de la voie 1

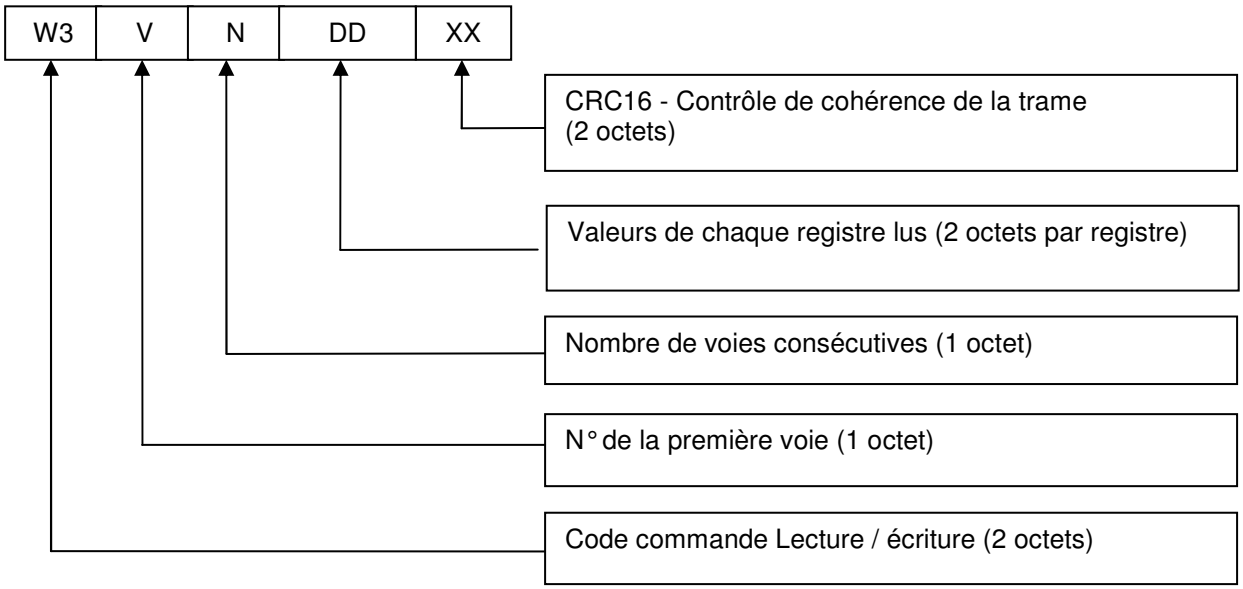
0x00 = Poids faible de la valeur à écrire dans le registre "LIMITE_COURANT " de la voie 2
0x00 = Poids fort de la valeur à écrire dans le registre " LIMITE_COURANT" de la voie 2

0x20 = Poids faible de la valeur à écrire dans le registre "LIMITE_COURANT" de la voie 3
0x4E = Poids fort de la valeur à écrire dans le registre "LIMITE_COURANT" de la voie 3

0x61 = Poids faible du CRC16
0x6E = Poids fort du CRC16

Code commande W3 - Format de la trame renvoyée par la main

Réponse à une demande d'écriture du registre "LIMITE_COURANT" de n voies consécutives et
Lecture du registre "POSITION_CODEUR" des n voies



Exemple : Réponse à une demande d'écriture des registres "LIMITE_COURANT" des voies 1 à 3
Et Lecture des registres "POSITION_CODEUR" des voies 1 à 3

Octets émis 0x57, 0x33, 0x01, 0x03, 0xE8, 0x03, 0x00, 0x00, 0x20, 0x4E, 0xBA, 0x7D

0x57 = Code ascii du caractère W
0x33 = Code ascii du caractère 3

0x01 = N° de la première voie
0x03 = Nombre de voies

0xE8 = Poids faible du registre "POSITION_CODEUR" de la voie 1
0x03 = Poids fort du registre "POSITION_CODEUR" de la voie 1

0x00 = Poids faible du registre "POSITION_CODEUR" de la voie 2
0x00 = Poids fort du registre "POSITION_CODEUR" de la voie 2

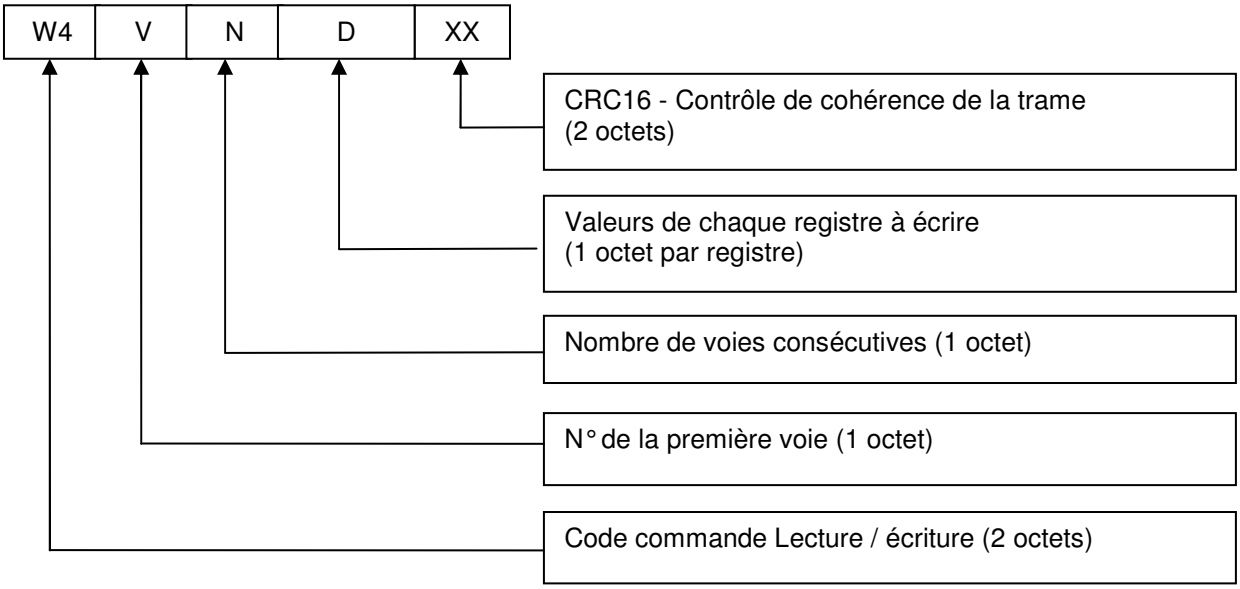
0x20 = Poids faible du registre "POSITION_CODEUR" de la voie 3
0x4E = Poids fort du registre "POSITION_CODEUR" de la voie 3

0xBA = Poids faible du CRC16
0x7D = Poids fort du CRC16

Attention : Pour ce type commande, la valeur de "POSITION_CODEUR" est retournée sur 2 octets et doit être considérée comme positive lorsqu'elle est inférieure à 60535 et négative lorsqu'elle est supérieure à cette valeur

Code commande W4 - Format de la trame émise par le PC

Demande d'écriture du registre "MODE_CMD_MOTEUR" de n voies consécutives et
Lecture du registre "VITESSE_MOTEUR" des n voies



Exemple : Demande d'écriture des registres "MODE_CMD_MOTEUR" des voies 1 à 3
Et Lecture des registres "VITESSE_MOTEUR" des voies 1 à 3

Octets émis 0x57, 0x34, 0x01, 0x03, 0x02, 0x01, 0x00, 0xA4, 0x65

0x57 = Code ascii du caractère W
0x34 = Code ascii du caractère 4

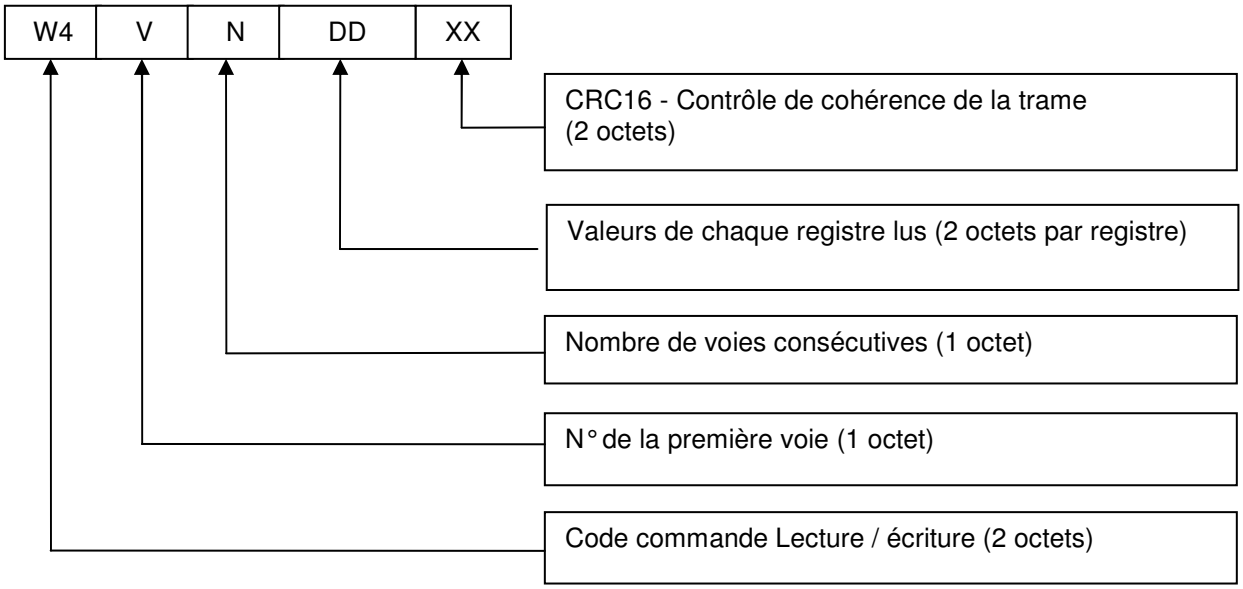
0x01 = N° de la première voie
0x03 = Nombre de voies

0x02 = Valeur à écrire dans le registre "MODE_CMD_MOTEUR" de la voie 1
0x01 = Valeur à écrire dans le registre "MODE_CMD_MOTEUR" de la voie 2
0x00 = Valeur à écrire dans le registre "MODE_CMD_MOTEUR" de la voie 3

0xA4 = Poids faible du CRC16
0x65 = Poids fort du CRC16

Code commande W4 - Format de la trame renvoyée par la main

Réponse à une demande d'écriture du registre "MODE_CMD_MOTEUR" de n voies consécutives et
Lecture du registre "VITESSE_MOTEUR" des n voies



Exemple : Réponse à une demande d'écriture des registres "MODE_CMD_MOTEUR" des voies 1 à 3
Et Lecture des registres "VITESSE_MOTEUR" des voies 1 à 3

Octets émis 0x57, 0x34, 0x01, 0x03, 0xE8, 0x03, 0x00, 0x00, 0x20, 0x4E, 0x9C, 0x4D

0x57 = Code ascii du caractère W
0x34 = Code ascii du caractère 4

0x01 = N° de la première voie
0x03 = Nombre de voies

0xE8 = Poids faible du registre "VITESSE_MOTEUR" de la voie 1
0x03 = Poids fort du registre "VITESSE_MOTEUR" de la voie 1

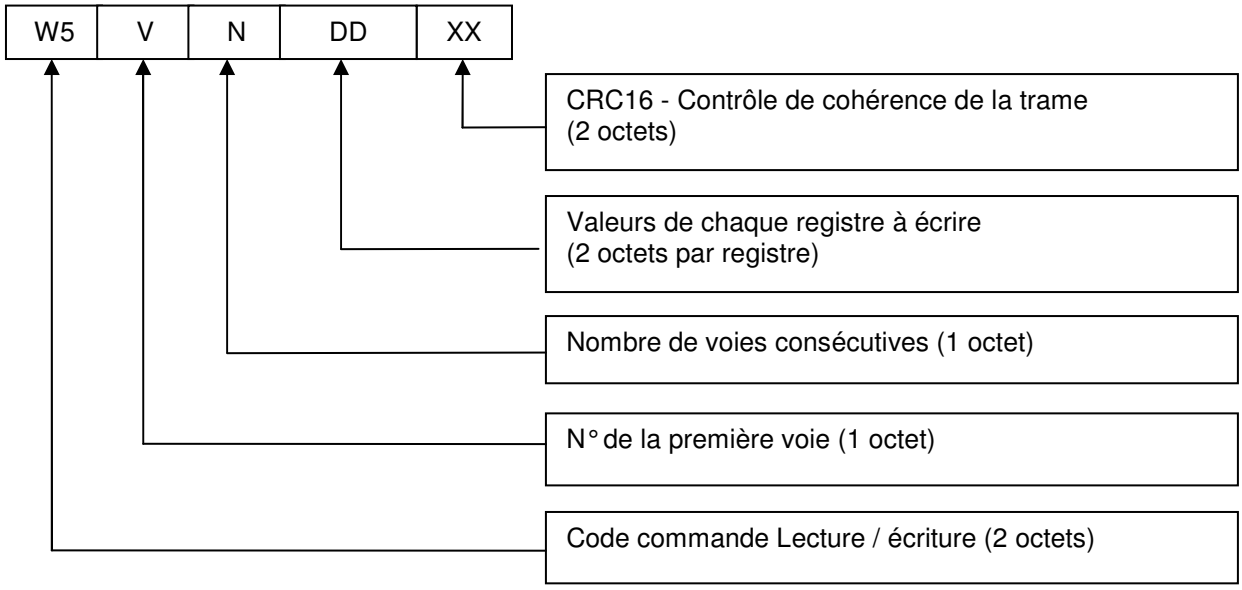
0x00 = Poids faible du registre "VITESSE_MOTEUR" de la voie 2
0x00 = Poids fort du registre "VITESSE_MOTEUR" de la voie 2

0x20 = Poids faible du registre "VITESSE_MOTEUR" de la voie 3
0x4E = Poids fort du registre "VITESSE_MOTEUR" de la voie 3

0x9C = Poids faible du CRC16
0x4D = Poids fort du CRC16

Code commande W5 - Format de la trame émise par le PC

Demande d'écriture du registre "CONSIGNE_TENSION_POSITION" de n voies consécutives et
Lecture du registre "VITESSE_MOTEUR" des n voies



Exemple : Demande d'écriture des registres "CONSIGNE_TENSION_POSITION" des voies 1 à 3
Et Lecture des registres " POSITION_CODEUR" des voies 1 à 3

Octets émis 0x57, 0x35, 0x01, 0x03, 0x20, 0x03, 0x00, 0x00, 0xF4, 0x01, 0x9F, 0x61

0x57 = Code ascii du caractère W
0x35 = Code ascii du caractère 5

0x01 = N° de la première voie
0x03 = Nombre de voies

0x20 = Poids faible de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 1
0x03 = Poids fort de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 1

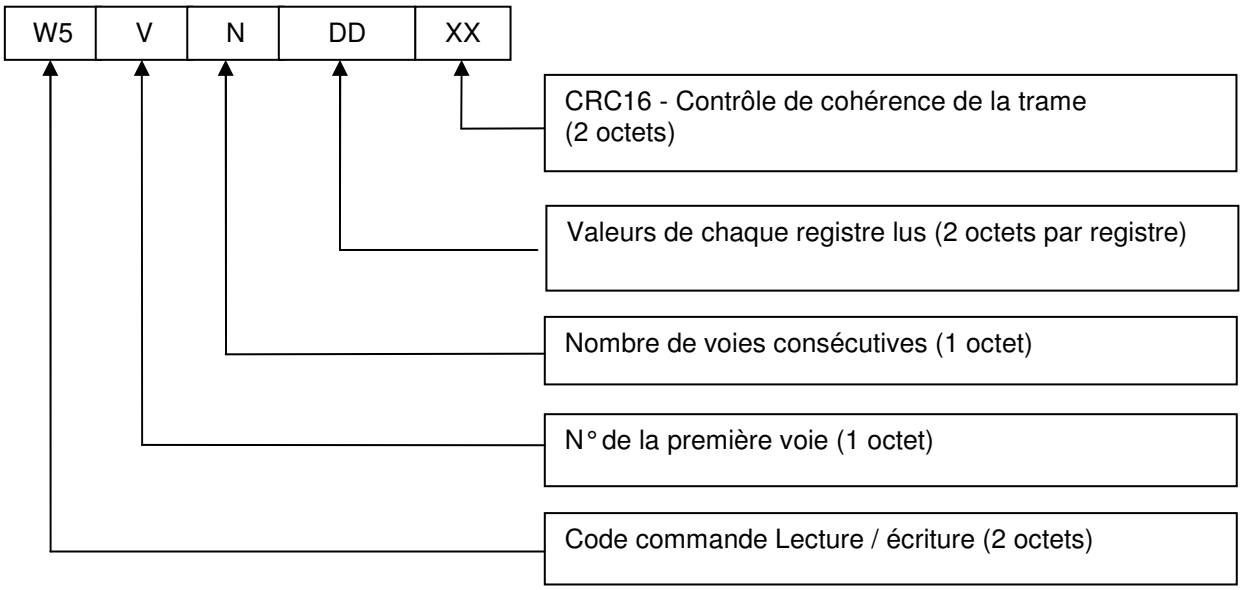
0x00 = Poids faible de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 2
0x00 = Poids fort de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 2

0xF4 = Poids faible de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 3
0x01 = Poids fort de la valeur à écrire dans le registre " CONSIGNE_TENSION_POSITION" de la voie 3

0x9F = Poids faible du CRC16
0x61 = Poids fort du CRC16

Code commande W5 - Format de la trame renvoyée par la main

Réponse à une demande d'écriture du registre "CONSIGNE_TENSION_POSITION" de n voies consécutives et lecture du registre "VITESSE_MOTEUR" des n voies



Exemple : Réponse à une demande d'écriture des registres "CONSIGNE_TENSION_POSITION" des voies 1 à 3 et lecture des registres "VITESSE_MOTEUR" des voies 1 à 3

Octets émis 0x57, 0x35, 0x01, 0x03, 0xE8, 0x03, 0x00, 0x00, 0x20, 0x4E, 0x91, 0xDD

0x57 = Code ascii du caractère W
0x35 = Code ascii du caractère 5

0x01 = N° de la première voie
0x03 = Nombre de voies

0xE8 = Poids faible du registre "VITESSE_MOTEUR" de la voie 1
0x03 = Poids fort du registre "VITESSE_MOTEUR" de la voie 1

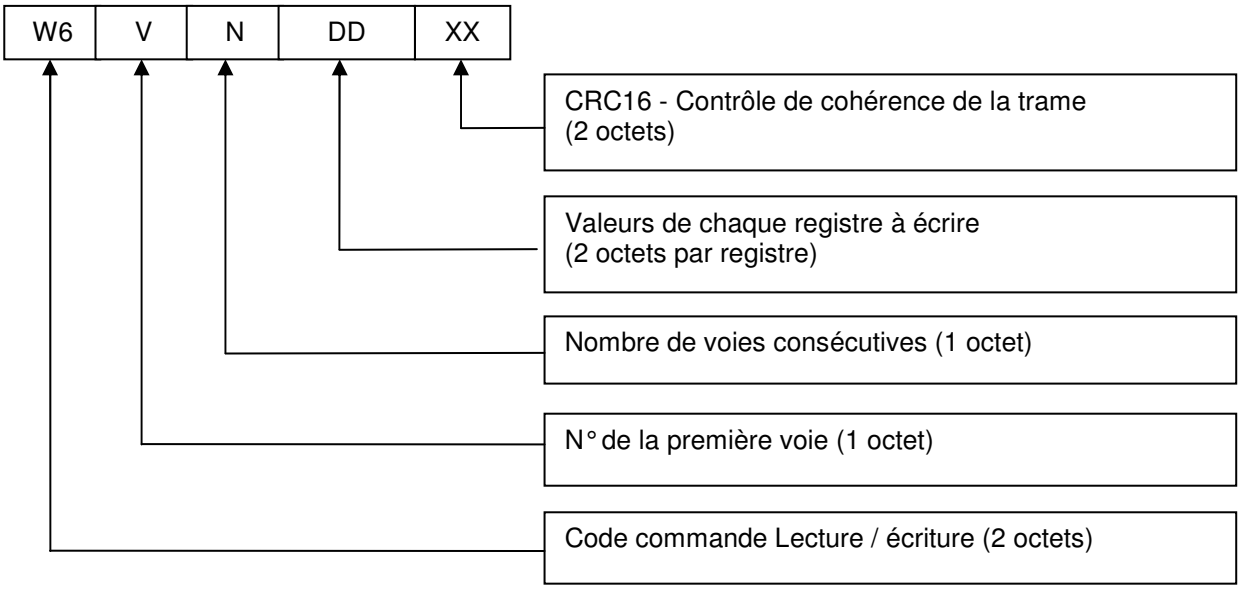
0x00 = Poids faible du registre "VITESSE_MOTEUR" de la voie 2
0x00 = Poids fort du registre "VITESSE_MOTEUR" de la voie 2

0x20 = Poids faible du registre "VITESSE_MOTEUR" de la voie 3
0x4E = Poids fort du registre "VITESSE_MOTEUR" de la voie 3

0x91 = Poids faible du CRC16
0xDD = Poids fort du CRC16

Code commande W6 - Format de la trame émise par le PC

Demande d'écriture du registre "LIMITE_COURANT" de n voies consécutives et
Lecture du registre "VITESSE_MOTEUR" des n voies



Exemple : Demande d'écriture des registres " LIMITE_COURANT" des voies 1 à 3
Et Lecture des registres "VITESSE_MOTEUR" des voies 1 à 3

Octets émis 0x57, 0x36, 0x01, 0x03, 0x30, 0x75, 0x00, 0x00, 0x20, 0x4E, 0x5E, 0x3E,

0x57 = Code ascii du caractère W
0x36 = Code ascii du caractère 6

0x01 = N° de la première voie
0x03 = Nombre de voies

0x30 = Poids faible de la valeur à écrire dans le registre "LIMITE_COURANT" de la voie 1
0x75 = Poids fort de la valeur à écrire dans le registre "LIMITE_COURANT " de la voie 1

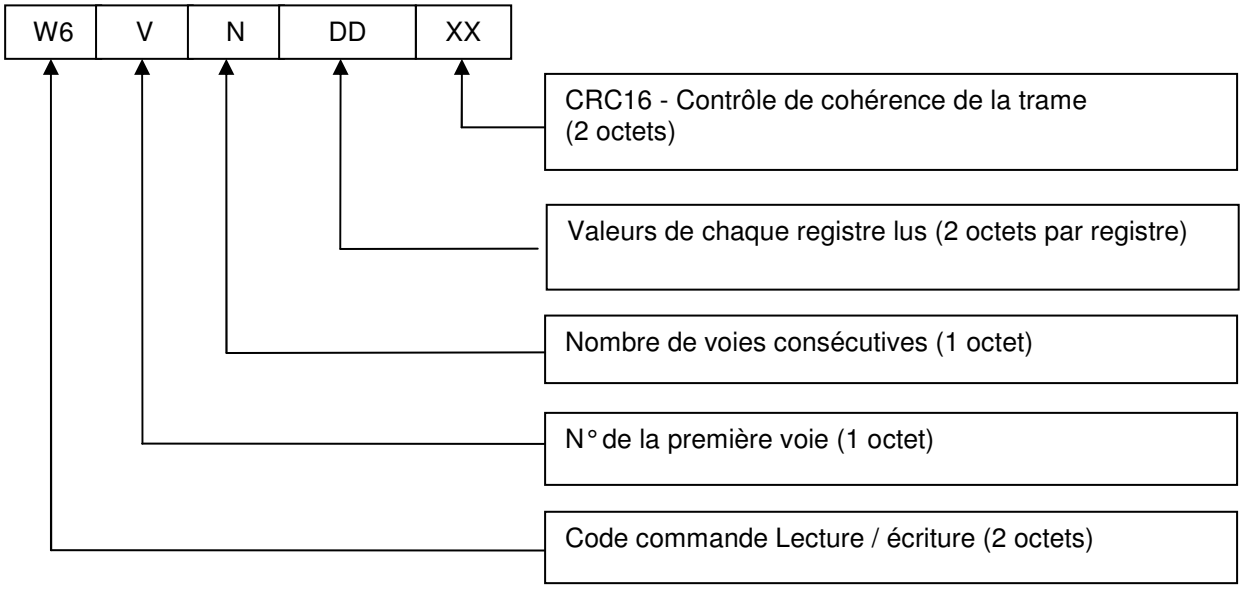
0x00 = Poids faible de la valeur à écrire dans le registre "LIMITE_COURANT " de la voie 2
0x00 = Poids fort de la valeur à écrire dans le registre " LIMITE_COURANT" de la voie 2

0x20 = Poids faible de la valeur à écrire dans le registre "LIMITE_COURANT" de la voie 3
0x4E = Poids fort de la valeur à écrire dans le registre "LIMITE_COURANT" de la voie 3

0x5E = Poids faible du CRC16
0x3E = Poids fort du CRC16

Code commande W6 - Format de la trame renvoyée par la main

Réponse à une demande d'écriture du registre "LIMITE_COURANT" de n voies consécutives et
Lecture du registre "POSITION_CODEUR" des n voies



Exemple : Réponse à une demande d'écriture des registres "LIMITE_COURANT" des voies 1 à 3
Et Lecture des registres "VITESSE_MOTEUR" des voies 1 à 3

Octets émis 0x57, 0x36, 0x01, 0x03, 0xE8, 0x03, 0x00, 0x00, 0x20, 0x4E, 0x85, 0x2D

0x57 = Code ascii du caractère W
0x36 = Code ascii du caractère 6

0x01 = N° de la première voie
0x03 = Nombre de voies

0xE8 = Poids faible du registre "VITESSE_MOTEUR" de la voie 1
0x03 = Poids fort du registre "VITESSE_MOTEUR" de la voie 1

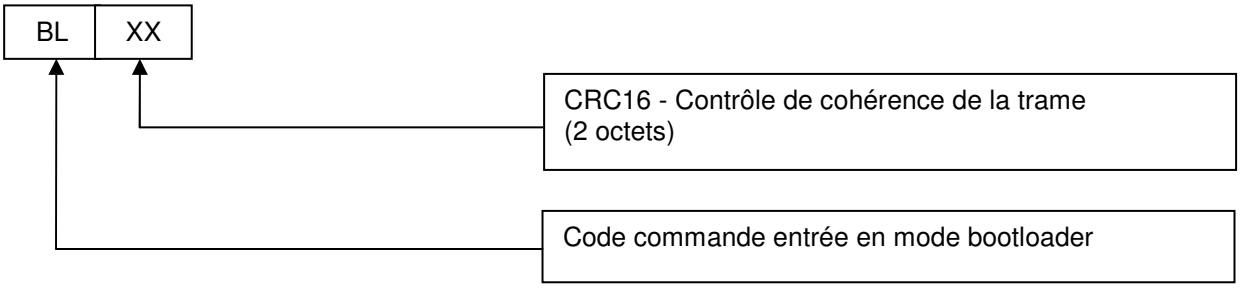
0x00 = Poids faible du registre "VITESSE_MOTEUR" de la voie 2
0x00 = Poids fort du registre "VITESSE_MOTEUR" de la voie 2

0x20 = Poids faible du registre "VITESSE_MOTEUR" de la voie 3
0x4E = Poids fort du registre "VITESSE_MOTEUR" de la voie 3

0x85 = Poids faible du CRC16
0x2D = Poids fort du CRC16

Code commande BL - Format de la trame envoyée par le PC

Demande d'entrée en mode de chargement d'un nouveau programme



Exemple : Demande d'entrée en mode de chargement d'un nouveau programme

Octets émis 0x42, 0x4C, 0x30, 0xE5

0x42 = Code ascii du caractère B
0x4C = Code ascii du caractère L

0x30 = Poids faible du CRC16
0xE5 = Poids fort du CRC16

Aucune réponse n'est retournée par la main à réception de ce code de commande.
Le microcontrôleur est directement configuré de manière recevoir un nouveau programme.
Le chargement du programme se fait à l'aide de l'utilitaire "Flash Magic" disponible sur Internet
Le PC doit impérativement libérer le port "COM" de manière à ce qu'il soit disponible pour "Flash Magic"

Calcul de CRC16 (méthode 1)

Cette méthode utilise peu de place en mémoire mais nécessite un temps d'exécution plus important que la méthode 2.

```
UVAR_16 CalculCrc16 (UVAR_8 *buf, UVAR_16 len )
{
    UVAR_8  byte_index, bit_index;
    UVAR_16 crc16 = 0xFFFF;

    for(byte_index = 0; byte_index < len; byte_index ++)
    {
        crc16 ^= *(buf + byte_index);

        for (bit_index = 0 ; bit_index < 8 ; bit_index ++ )
        {
            if(crc16 & 0x0001) crc16 ^= 0xA001;
            crc16 >>= 1;
        }
    }
    return crc16;
}
```

Calcul de CRC16 (méthode 2)

Cette méthode utilise plus de place en mémoire mais nécessite un temps d'exécution moins important que la méthode 1.

```
/* Table of CRC values for high–order byte */
const UVAR_8 TABLE_CRC_HI[] = {
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
0x81,0x40 } ;
```

```
/* Table of CRC values for low–order byte */
const UVAR_8 TABLE_CRC_LO[] = {
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7, 0x05, 0xC5, 0xC4,
0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD,
0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32, 0x36, 0xF6, 0xF7,
0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE,
0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1, 0x63, 0xA3, 0xA2,
0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB,
0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0, 0x50, 0x90, 0x91,
0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88,
0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83, 0x41, 0x81, 0x80,0x40 };



UVAR_16 APP_Crc16_Calc( UVAR_8 *buffer, UVAR_32 len)
{
   UVAR_8 crc_hi = 0xFF ; /* high byte of CRC initialized */
   UVAR_8 crc_lo = 0xFF ; /* low byte of CRC initialized */
   UVAR_8 index = 0; /* will index into CRC lookup table */
   UVAR_16 crc;
   while (len--)
   {
        index = crc_lo ^ *buffer++ ;
        crc_lo = crc_hi ^ TABLE_CRC_HI[index] ;
        crc_hi = TABLE_CRC_LO[index] ;
    }
    crc = crc_hi;
    crc = (crc * 256) + crc_lo;
    return crc ;
}
```