Identification des trames TCP

L'échange de données entre le panel PC et notre PC s'effectue de manière continue. Les données qui transitent sur le réseau CAN, RS 485 et USB sont centralisées sur le panel PC qui les envoie en continu au PC via la liaison TCP/IP. Le protocole de communication n'est pas de type requête/réponse ce qui permettrait de faire correspondre les valeurs que l'on modifie sur le système avec celles que l'on reçoit. Cela complique alors la rétro-engineering.

Les trames sont enregistrées sur Wireshark selon plusieurs tests isolant les communications. D'abord aucune communication n'est reliée au panel PC (ni CAN, ni RS485, ni USB), puis uniquement la liaison RS485, uniquement la liaison USB et uniquement la liaison CAN.

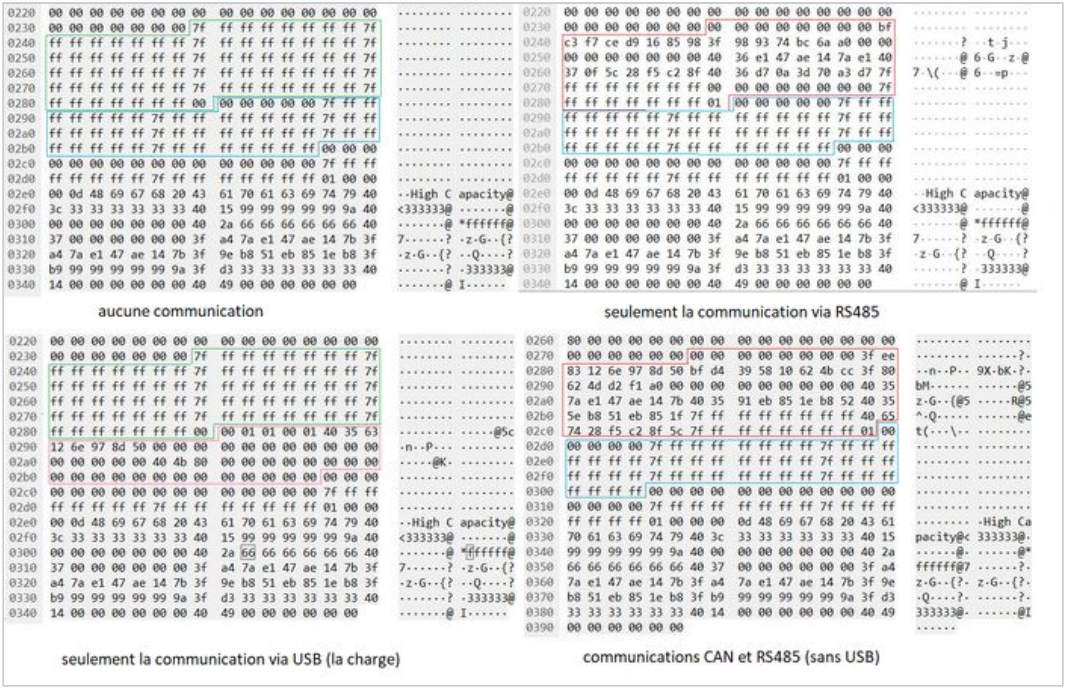
Premier cas, trames du PC vers panel PC (IP 192.168.0.2 vers IP 192.168.0.1)

Dans les trois tests : trames de longueur 54 octets, aucun octet de données.

Deuxième cas, trames du panel PC vers PC = les données (IP 192.168.0.1 vers IP 192.168.0.2)

- pour aucune communication, uniquement RS485 ou uniquement USB, on a une longueur de 847 avec 793 octets de données.
- pour seulement CAN, une longueur de 947 avec 893 octets de données

On remarque que sur les 20 dernières lignes de paquets, on arrive à trouver des similitudes, et notamment à isoler les paquets associés à la communication RS et à la communication USB : c'est ce que montrent les captures ci-dessous (entre temps un test avec la communication CAN et RS485 a été réalisé pour appuyer nos résultats).



paquets TCP/IP lus sur Wireshark

La correspondance de ces différents paquets d'octets en code ASCII ne donne rien d'exploitable. En plus d'associer les valeurs au variables, il faut également trouver le mode de conversion utilisé pour retrouver les valeurs réelles indiquées par l'interface homme-machine. On remarque également que la dernière partie de la trame où l'on voit indiqué "High Capacity" dans la correspondance en ASCII correspond aux valeurs de configuration des batteries (état de charge, profondeur de charge) qui sont contenues dans un fichier sur le panel PC mais uniquement les indicatifs nominaux sont remarquables dans cette trame.

Chaque communication va maintenant être analysée plus en détails.

- Partie CAN

Un test est réalisé sur la trame x110 qui ne varie quasiment pas et notamment sur la température de la pile : on recherche la valeur 07F9 (vu sur NI XNET Monitor) qui correspond à 20,41°C.
Dans les trames TCP on ne retrouve pas de 07. Les trames s'échangent bien toutes les 200ms soit une période plus longue que celle d'envoie de la trame x110 sur le CAN. Les données de celle-ci devraient donc apparaître.
A la vue du nombre de données transitant sur le CAN (plus d'une vingtaine), un reverse engineering ne semble pas approprié, d'autant plus que ces variables ne peuvent pas être isolées.

Une documentation étant disponible et décrivant le contenu de toutes les trames transitant sur le CAN, c'est donc l'étude de cette dernière solution qui est privilégiée. D'autant plus que le système contient un port CAN non utilisé en fin de chaîne. Il a donc été très facile de lire cette communication depuis un sniffer CAN tout en comparant les valeurs reçues sur le panel PC. Dans un second temps, une carte NI instrument a été achetée afin de lire les données sur le PC, d'abord sur les modules NI XNET Bus Monitor (vérification de la réception des trames) puis sur Labview avec de la programmation.

- Partie RS

Dans cette partie, uniquement la communication RS485 a été connectée au panel PC afin d'identifier les trames TCP/IP associées. Comme le montrent les tests précédents, On suppose que les 5 lignes suivantes correspondent aux données du RS485 soit 80 octets de données sur les 793 qui varient uniquement lorsque le bus RS485 est connecté au système.
Le logiciel DCON présent sur le panel PC permet de contrôler quel canal analogique du module ICPDAS transmet des données sur le bus RS485, et donc d'isoler les variables voulues.

```
0230   00 00 00 00 00 00 00 7f ff ff ff ff ff ff ff 7f   ................
0240   ff ff ff ff ff ff ff 7f ff ff ff ff ff ff ff 00   ................
0250   00 00 00 00 00 00 00 40 32 f3 33 33 33 33 33 7f   .......@2.33333.
0260   ff ff ff ff ff ff ff 7f ff ff ff ff ff ff ff 7f   ................
0270   ff ff ff ff ff ff ff 7f ff ff ff ff ff ff ff 7f   ................
0280   ff ff ff ff ff ff ff 01 00 00 00 00 00 7f ff ff   ................
```

paquets TCP/IP associés aux données RS485 avec uniquement la température S1

En isolant une variable sur les sept que doit transmettre le bus RS485, on a donc pu se placer dans une configuration où le système ne transmet qu'une valeur à l'interface. Cette valeur était une valeur de température de 21.1°C. Chaque variable a été isolée afin de retrouver l'emplacement des paquets associés. L'ensemble des variables peut alors être repris pour continuer l'analyse.

```
0230   00 00 00 3f ea d0 e5 60 41 89 37 40 03 5c 28 f5   ...?...`A.7@.\(.
0240   c2 8f 26 bf a4 7a e1 47 ae 27 f0 00 00 00 00 00   ..&..z.G.'......
0250   00 00 00 40 35 0f 5c 28 f5 c2 8f 40 35 51 eb 85   ...@5.\(...@5Q..
0260   1e b8 52 40 35 19 99 99 99 99 9a c0 78 eb 85 1e   ..R@5.......x...
0270   b8 51 ec 40 62 da e1 47 ae 14 7a 7f ff ff ff ff   .Q.@b..G..z.....
0280   ff ff ff 01 00 00 00 00 00 7f ff ff ff ff ff ff   ................
```

paquets TCP/IP associés aux données RS485

Les paquets suivants, car intéressants, vont être analysés :

Couleur	Variable	Valeur HEL	Paquet
VERTS	TS1	21,1 °C	40 35 0F 5C 28 F5 C2 8F
	TS2	21,1 °C	40 35 51 EB 85 1E B8 52
	TS3	21,1 °C	40 35 19 99 99 99 99 9A
JAUNE	H2 Flow	0 Nl/min	00 00 00 00 00 00 00 00
ORANGE	I AC	-2,5 A	BF A4 7A E1 47 AE 27 F0
ROUGE	I Op	0,1 A	40 03 5C 28 F5 C2 8F 26

Les valeurs de températures se ressemblent énormément, ce qui est rassurant puisqu'elles ont, à 0,01 près, la même valeur sur le panel PC. Cependant, la valeur de courant I Op a un paquet assez similaire à celui de la température TS1 alors que la valeur réelle est complètement différente.
Une autre différence flagrante est présente entre le paquet du débit H2 flow et celui du courant I AC. Alors que les valeurs réelles - sans prendre en compte l'unité - sont très proches (0 contre 0,1), les paquets sont radicalement opposés.

Une nette difficulté de compréhension des paquets se note.
Pour pousser l'analyse jusqu'au bout, les octets de fin ne correspondent pas à un checksum (avec ou sans le premier octet, que l'on pourrait éliminer car caractère de début de chaîne). La conversion de hexadécimal vers décimal, avec ou sans les octets de début et fin, ne

correspond à aucune valeur cohérente pour les différentes variables. Enfin, une conversion de type IEEE-754 Floating Point Converter ne présente pas de similitudes avec les valeurs réelles non plus.

Un reverse engineering reste donc compliqué et des tests beaucoup plus poussés doivent être menés pour atteindre les résultats espérés.
Entre temps, la communication RS485 avec le PC a été établie. Le fait de doubler les communications a été abandonné pour simplement remplacer le panel PC. Un premier test avec le logiciel DCON sur le PC a permis de voir que la connexion était établie. Ensuite, grâce à de la documentation trouvée sur internet sur le protocol DCON, l'envoie de requête de lecture ainsi que la réception des données avec analyse de celles-ci a été vérifiée sur le terminal PuTTy. Enfin, la liaison sur Labview a été réalisée.

- Communication USB/Port COM virtuel : la charge

Plusieurs tests sont réalisés pour isoler les variables. Ayant des problèmes de convertisseurs, très peu de variation des valeurs U, I, P peuvent être effectuées sur la charge si celle-ci est connectée au panel PC et ses valeurs sont donc à 0 pour les six premiers tests. Nous avons tout de même réussi à changer U sur les deux derniers tests.

test 1 : remote on, input off, CP

```
00 01 01 00 01 3f 9f be    76 c8 b4 39 58 3f 70 62
4d d2 f1 a9 fc 00 00 00    00 00 00 00 00 40 4b 80
00 00 00 00 00 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00
```

test 2 : remote on, input off, CC

```
00 01 01 00 00 3f 9f be    76 c8 b4 39 58 3f 70 62
4d d2 f1 a9 fc 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00 40 97 70    00 00 00 00 00 00 00 00
00 00 00 00 00
```

test 3 : remote on, input on, CP

```
01 01 01 00 01 3f 9f be    76 c8 b4 39 58 3f 70 62
4d d2 f1 a9 fc 00 00 00    00 00 00 00 00 40 4b 80
00 00 00 00 00 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00
```

test 4 : remote on, input on, CC

```
01 01 01 00 00 3f 9f be    76 c8 b4 39 58 3f 70 62
4d d2 f1 a9 fc 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00 40 97 70    00 00 00 00 00 00 00 00
00 00 00 00 00
```

test 5 : remote off, input off, CC

```
00 01 00 00 00 3f 9f be    76 c8 b4 39 58 3f 70 62
4d d2 f1 a9 fc 00 00 00    00 00 00 00 00 40 4b 80
00 00 00 00 00 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00
```

test 6 : remote off, input off, CP

```
00 01 00 00 01 3f 9f be    76 c8 b4 39 58 3f 70 62
4d d2 f1 a9 fc 00 00 00    00 00 00 00 00 40 4b 80
00 00 00 00 00 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00
```

test 7 : remote on, input off, CC, I=0A, V=22V, P=0W

```
00 01 01 00 01 40 36 01    89 37 4b c6 a8 3f a6 04
18 93 74 bc 6a 00 00 00    00 00 00 00 00 40 4b 80
00 00 00 00 00 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00
```

test 8 : remote on, input off, CC, I=0A, V=0,1V, P=0W

```
00 01 01 00 00 3f e8 9b    a5 e3 53 f7 cf 3f 94 7a
e1 47 ae 14 7b 00 00 00    00 00 00 00 00 40 4b 80
00 00 00 00 00 00 00 00    00 00 00 00 00 00 00 00
00 00 00 00 00
```

paquets TCP/IP associés aux données USB (charge)

Comme le montrent les caractères surlignés, des ressemblances entre les tests permettent de retrouver certaines valeurs, notamment le mode Remote (on/off) ainsi que Input (on/off) et le mode de contrôle (CP contrôle en puissance, CC contrôle en courant). En surlignage jaune est indiqué les paquets qui ne suivent finalement pas toujours ce que l'on peut noter dans les quatre premiers tests. Les indicateurs de mode d'utilisation ont donc l'air d'être facilement disponibles dans ces paquets.

Les valeurs de U, I et P maintenant. Comme mentionné plus tôt, seule la valeur de U a pu varier lors de ces tests. Cependant, c'est l'ensemble ci-contre en rose (sur chaque trame) qui varie.

00 01 01 00 01 40 36 01 89 37 4b c6 a8 3f a6 04
18 93 74 bc 6a 00 00 00 00 00 00 00 00 40 4b 80
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00

test 6 (0V)	3f 9f be 76 c8 b4 39 58 3f 70 62 4d d2 f1 a9 fc
test 7 (22V)	40 36 01 89 37 4b c6 a8 3f a6 04 18 93 74 bc 6a
test 8 (0,1V)	3f e8 9b a5 e3 53 f7 cf 3f 94 7a e1 47 ae 14 7b

La conversion de ces trames en décimal n'aboutit à aucun résultat cohérent, avec ou sans les premiers caractères et/ou les caractères de fin. La première valeur (surlignée en bleu) est bien égale pour 0 et 0,1V mais est plus faible avec la tension qui augmente. De la même manière qu'avec le RS485, différents types de conversion sont essayés mais aucune ne donne lieu à une logique entre les trois tests. Enfin, des essais avec le type de conversion utilisée par le module USB IF-U1 (pourcentage selon une valeur nominale de courant) ont été réalisés mais toujours sans succès.

La conclusion reste la même que pour le RS485. Alors que les paquets associés aux variables transmises sont facilement identifiables, l'analyse des paquets montre une nécessité de connaître plus amplement le type de conversion utilisé par HEL, qui n'est pas divulgué par l'entreprise.

En revanche, en utilisant un module sniffer USB entre le panel PC et la charge, des trames correspondants à la documentation du module USB IF-U1 sont identifiables et cette fois les valeurs réelles sont retrouvées. Grâce à un logiciel de lecture et écriture série, l'envoie de requêtes est possible et la réception des données également. Après une étude poussée de la documentation, les différentes requêtes sont comprises ce qui permet de réaliser le programme Labview associé. Cependant, pour les mêmes raisons que la liaison RS485, le flux de données n'est pas doublé à la différence de la liaison CAN. De ce fait, pour communiquer avec la charge sur Labview il faut déconnecter cette dernière du panel PC.