La bibliothèque DCCpp_S88 pour Arduino

Basée sur le logiciel libre DCC++ de Greg E. Berman puis repris par Thierry Paris qui en a fait une bibliothèque DCCpp, j'ai ajouté la retro-signalisation S88 initiée par Xavier Bouillard ainsi que quelques fonctions secondaires pour produire la bibliothèque DCCpp_S88.

Cette bibliothèque DCCppS88 reste compatible avec ses précédentes et le format de commande DCC. J'utilise cette bibliothèque avec un Arduino MEGA2560 qui donne toute satisfaction à l'usage d'un modéliste. Cette paire dispose d'une bonne stabilité éprouvée qui constitue une très bonne base pour piloter un réseau ferroviaire DCC. Je déconseille l'utilisation d'un Uno ou d'un Nano.

La version de base utilise l'interface USB du MEGA2560. Avec l'ajout d'une interface Ethernet offrant un connecteur RJ45, il devient possible de brancher l'ensemble sur un réseau Ethernet domestique avec une Box Ethernet ou un routeur. Il existe aussi d'autres extensions en WiFi.

La bibliothèque **DCCppS88** pour Arduino donne pleinement satisfaction à l'usage donc je l'ai utilisée et j'ai construit une maquette. Ensuite avec SuperN, nous avons conçu un circuit imprimé pour faciliter le câblage du circuit intégré AM26C31 et les divers connecteurs dont nous avons besoin pour relier le MEGA2560 au différents Booster et aux modules de lecture du courant.

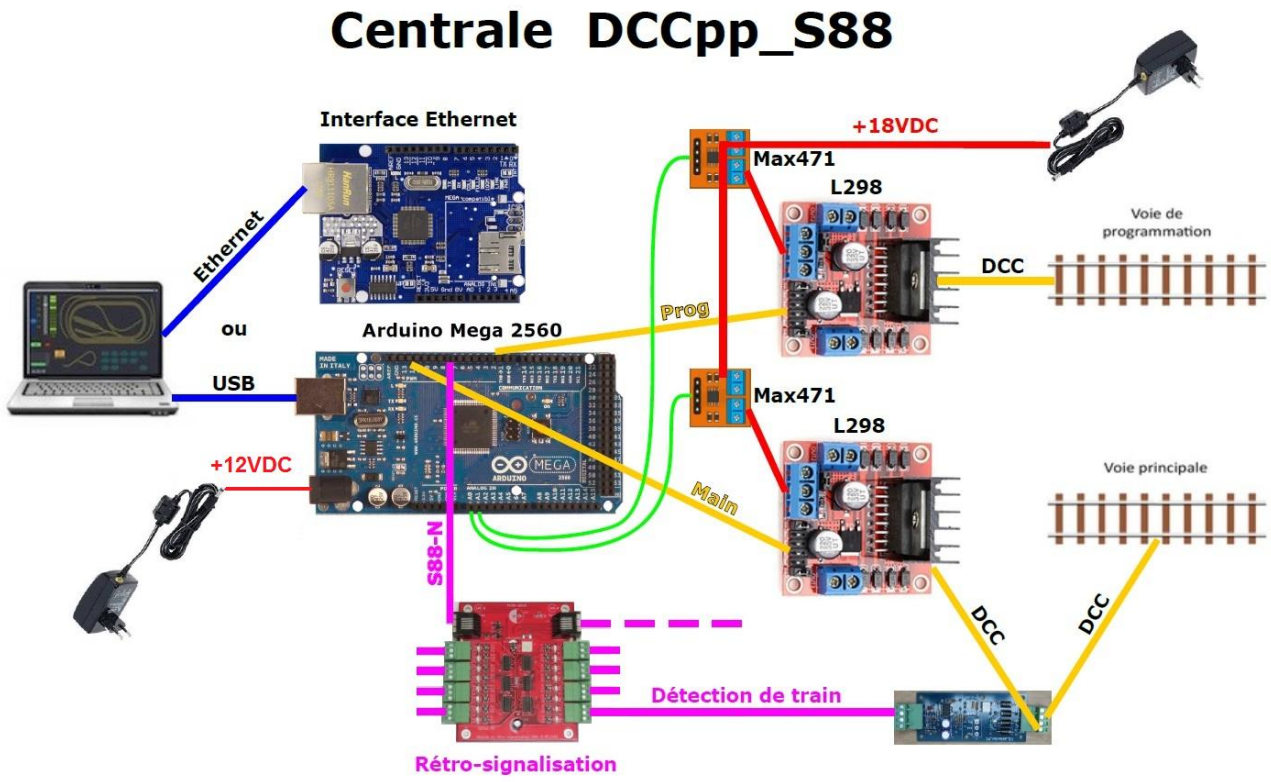

Figure 1: Synoptique simplifié de la centrale DCCpp_S88

Un circuit imprimé a été réalisé pour simplifier le câblage par le modéliste.

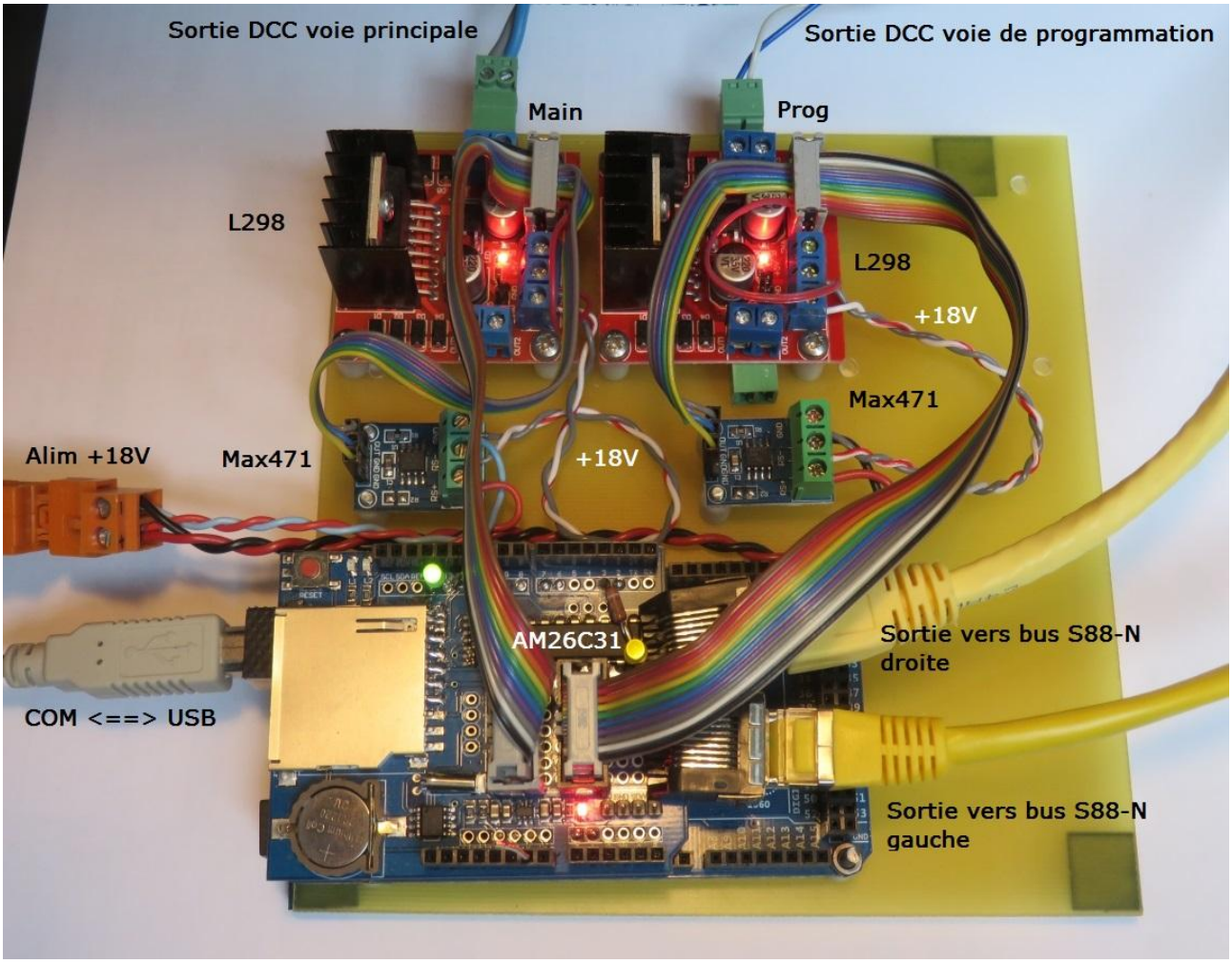

Figure 2 : La centrale DCC avec retro-signalisation S88, interface USB ou Ethernet

Note : pour pouvoir se connecter à Ethernet, il suffit de placer l'interface Ethernet entre le MEGA2560 et le circuit imprimé qui contient le câblage des connecteurs. Ci-dessus le circuit intégré AM26C31 et les divers connecteurs sont câblés sur une carte de prototypage au même format et posée sur le MEGA.

La bibliothèque **DCCppS88** permet de piloter plusieurs locos en même temps, en adresse courte, longue et UM, offre une sortie DCC vers les voies principales et une sortie DCC vers une voie de programmation indépendante. La commande se fait par l'intermédiaire de fenêtres sur l'écran du PC/Tablette/Smartphone en WiFi qui communiquent avec le MEGA.

Je précise que cette bibliothèque peut lire 2 bus **S88** de 256 capteurs chacun avec 2 entrées RJ45 : DataL et DataR. Ce sont 2 ports S88 distincts pour simplifier le câblage des grands réseaux. Cette fonction **S88-N** a été testée avec un module RM-GB-8-N de Littfinski DatenTechnik et avec mes modules S88-N-16E décrit dans la page "Rétro-signalisation". La même fonction **S88** a été intégrée dans DCC++ par Xavier avec qui je l'ai co-écrite.

Cette bibliothèque DCCppS88 avec la lecture du bus de rétro-signalisation S88 est compatible avec les logiciels de commande de réseaux ferroviaires tel que WDD6, CDT31, DMC, Cabine, control, JMRI, CDM-Rail, Rocrail, etc...

Hardware

J'utilise un Arduino **Mega2560** sur lequel je place une carte fille pour câbler un driver RS485/7, SN75LBC176 ou AM26C31 qui transforme la sortie PWM en 2 polarités inverses afin d'alimenter les modules Booster L298N. L'Arduino MEGA est connecté avec TX0/RX0 à l'interface USB ou a l'interface Ethernet pour être relié au PC qui contient les logiciels de pilotage des trains.

Une alimentation 19VDC/3A (ou plus) alimente les Boosters L298N qui fournissent le DCC, lesquels alimentent les rails. On peut aussi utiliser des Boosters LMD18200 qui ne nécessitent pas de AM26C31 mais ils sont plus chers. Une alimentation de PC portable suffira dans la plupart des cas. On peut aussi utiliser un chargeur indépendant de 9 ou 12VDC pour alimenter la carte MEGA WiFi. J'ajoute des modules MAX471 pour mesurer le courant sur les voies de programmation et principales. Coté pratique on doit connecter quelques fils et percer quelques trous pour assembler tous les éléments avec des vis afin de les solidariser ensemble.

Voici les principaux composants utilisés pour construire cette centrale DCC :

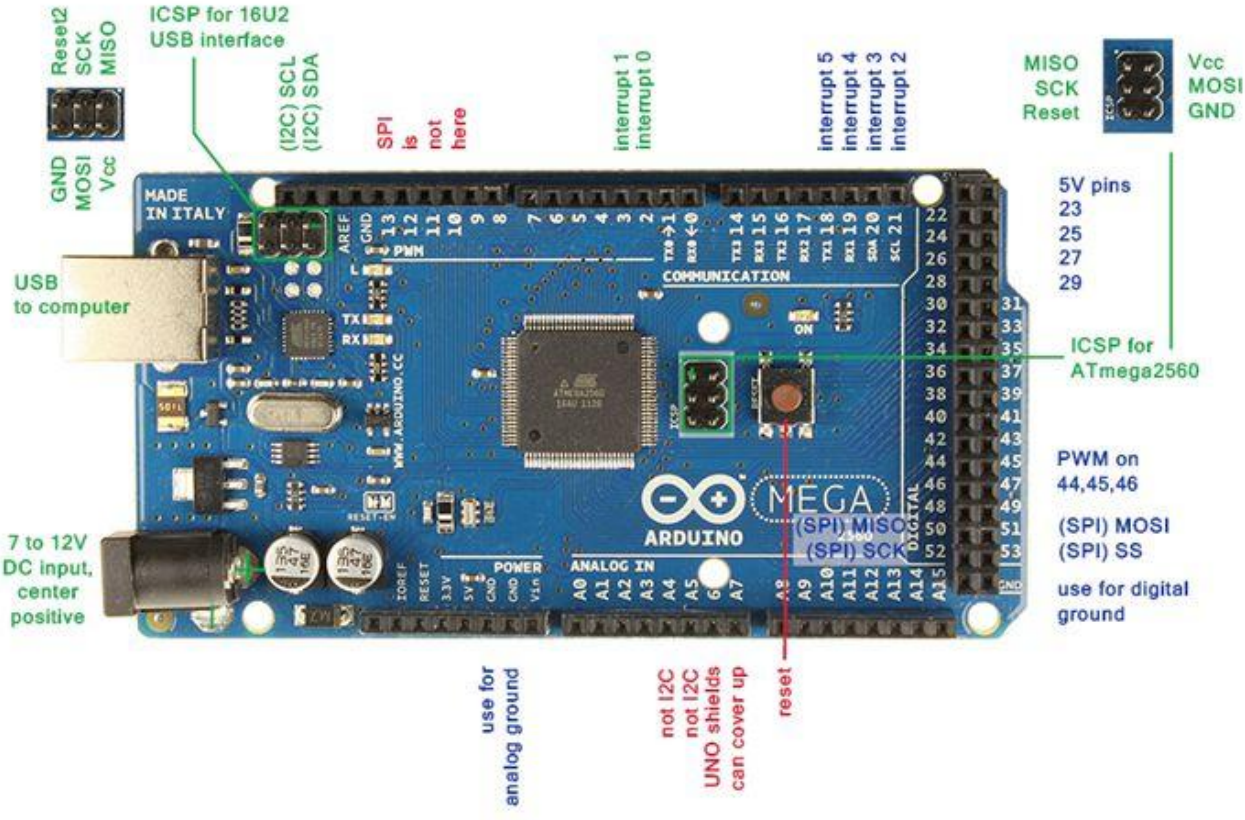

Figure 3 : Arduino MEGA 2560

Figure 4 : L293N

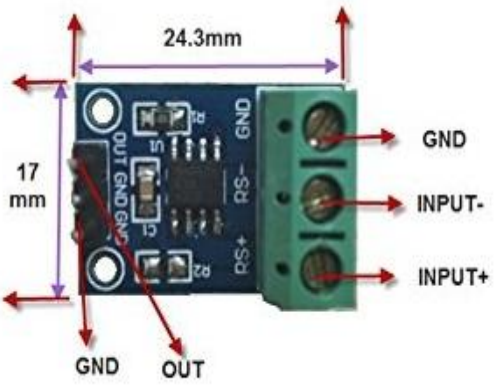

Figure 5 : Max471

Attention : Certains MAX471 sont défectueux. Avant de les connecter, vérifier avec un multimètre que la résistance lue entre RS+ et RS- soit proche de zéro Ohms. Sinon vous risquez d'endommager le MEGA.

Plus 2 connecteurs RJ45 pour le bus S88-N :

Adieu aux forêts de fils qui partent de l'Arduino, un simple câble RJ45 suffit. Les infos des capteurs de position des trains sur les cantons seront rassemblées sur place par une carte de rétro-signalisation. Puis elles sont transmises par le câble RJ45 vers la carte suivante et ainsi de suite le long du réseau jusqu'au MEGA. Pas besoin de boitier d'interface supplémentaire, tout est inclus. Le câble RJ45 du bus S88-N télé-alimente aussi les cartes, donc pas besoin d'alimenter séparément les cartes de rétro-signalisation. Sur le bus S88-N, le logiciel permet le mélange des cartes à 8 entrées avec celles à 16 entrées.

Note : au lieu d'utiliser le booster L298N, il est possible d'utiliser le booster LMD18200 en adaptant son câblage :

MEGA		LMD18200 (sans utiliser le AM26C31)
GND pin 14	==>	GND
DCC pin 12	==>	DIR
EN pin 3	==>	PWM
GND pin 14	==>	BRAKE

Schémas du montage

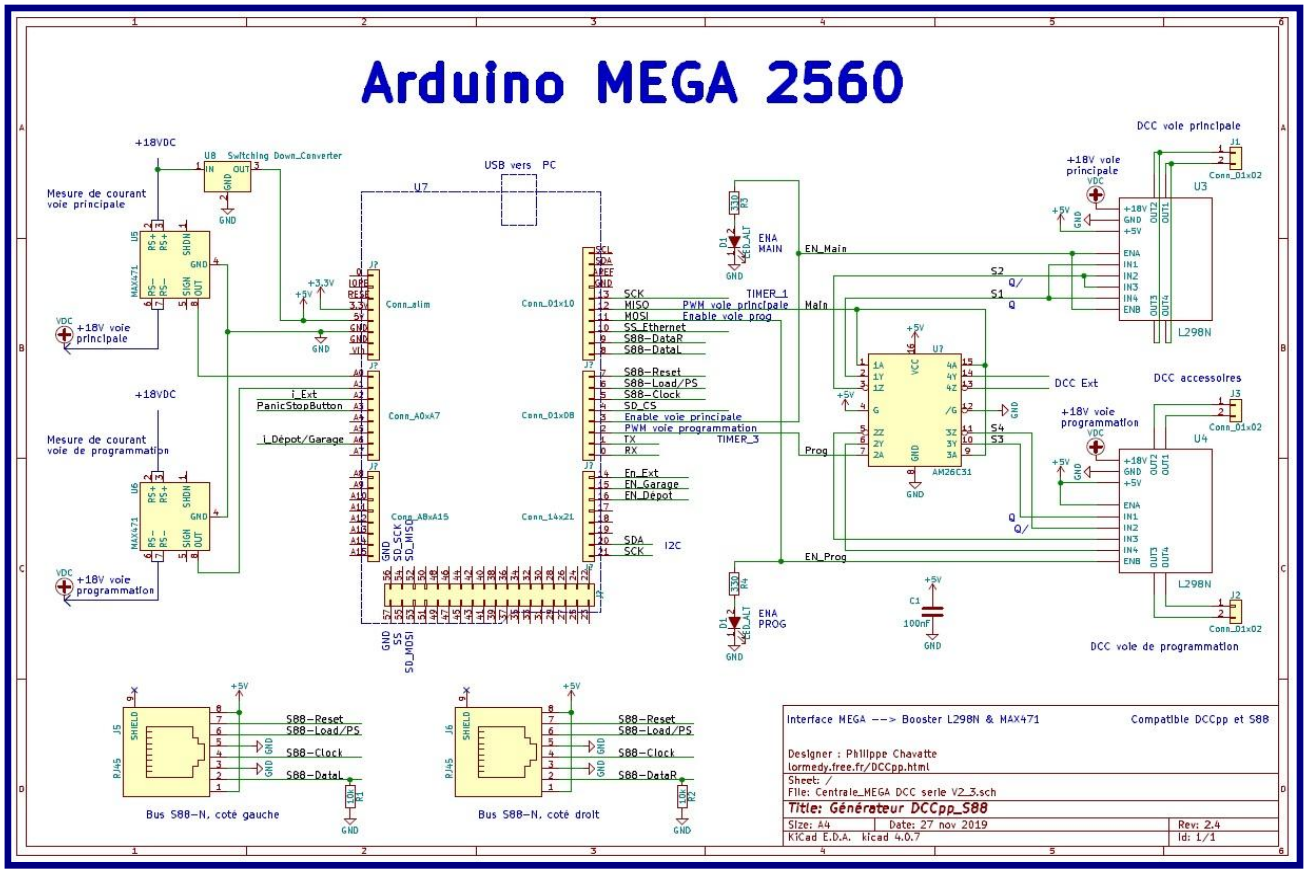

Figure 6: Plan de câblage de la centrale DCCpp_S88

Firmware Arduino (Commande DCC)

J'installe le logiciel DCCpp avec option **S88** qui se télécharge ici : **la biliothèque DCCppS88 pour Arduino** et j'ouvre l'IDE d'Arduino. Ensuite j'édite le fichier DCCppS88.ino pour vérifier la configuration avec l'interface série de l'Arduino à 115200 bauds. Choisir les interfaces utilisées indiquées par les commentaires encadrés par *****.

Enfin je compile les fichiers pour un MEGA2560 et je le téléverse. J'obtiens :

Rapport de compilation pour un Arduino Mega 2560 :

COMM Série/USB:

```
- Le croquis utilise 20246 octets (7%) de l'espace de stockage de programmes. Le
maximum est de 253952 octets.
- Les variables globales utilisent 711 octets (9%) de mémoire dynamique, ce qui
laisse 7481 octets pour les variables locales. Le maximum est de 8192 octets.
```

COMM Ethernet:

```
Le croquis utilise 39164 octets (15%) de l'espace de stockage de programmes. Le
maximum est de 253952 octets.

Les variables globales utilisent 1781 octets (21%) de mémoire dynamique, ce qui
laisse 6411 octets pour les variables locales. Le maximum est de 8192 octets.
```

Je connecte l'Arduino sur une entrée USB du PC qui sera automatiquement redirigé vers un port COMx et détecté par l'IDE Arduino. Après la compilation, je téléverse le code dans l'Arduino et j'ouvre le Moniteur série associé à l'IDE Arduino. Comme j'ai activé le DEBUG, je lis le rapport d'initialisation envoyé par l'Arduino et je peux déjà envoyer des commandes manuelles vers les locos qui réagissent immédiatement. Puis je ferme le Moniteur pour libérer le port COM. Important !

Logiciel

Cette centrale DCC accepte toutes les commandes compatibles générées par DCC++, DCCpp et DCCppS88. La suite du projet se trouvera dans le menu "Logiciel" du site Internet.

On peut commander ce montage de plusieurs manières : soit avec du logiciel résident sur le PC quelque soit l'interface, soit avec du logiciel enregistré sur la carte SD qui se trouve sur l'interface Ethernet.

Utilisation avec l'interface Ethernet :

A cet effet un logiciel "**control.htm**" ainsi que le fichier des locos "**locos.jso**" se trouve dans un répertoire nommé a_copier_sur_carte_SD qui porte bien son nom. Après les avoir enregistrés sur la carte SD et placé celle-ci dans son lecteur sur l'interface Ethernet, il suffit de taper l'adresse IP dans son navigateur pour ouvrir le programme de commande DCC qui ira lire le fichier des locos qu'on aura préalablement enregistré. Ex : 192.168.0.200:2560 et ainsi on peut commander ses locos à partir de la centrale DCC.

En seconde option, dans ce même répertoire, on peut lancer directement à partir du PC le programme "**controlDCC.html**" lequel ira lire le fichier des locos "**locos.js**" plus important en taille et qui est commun avec les logiciels conçus pour le **centrale DCC en WiFi.**

Utilisation avec l'interface série/USB :

Le logiciel CDT31 résident sur le PC permet de commander ses trains en DCC à partir de la liaison USB. Il nécessite d'utiliser Node.js qu'il faut installer aussi sur le PC.

Enfin on peut utiliser des logiciels compatible DCC++ ou DCCpp tel que CDM-Rail pour conduire ses trains avec les 2 types d'interface.

!!! Ces programmes utilisent des liaisons Ethernet uniquement avec le protocole HTTP.

Rétro-signalisation S88

J'ai incorporé le programme maitre **S88** dans la bibliothèque **DCCppS88** décrite ici. En conséquence j'utilise un SEUL Arduino MEGA pour gérer tout mon réseau DCC : un seul et unique ! La rétro-signalisation est totalement intégrée dans la bibliothèque **DCCppS88** sans ralentir l'Arduino. Une trame de 512 bits maxi est lue en permanence par l'Arduino sur le bus **S88-N** en moins de 60ms.

Les données des capteurs **S88** sont lues sur 2 entrées simultanément pour ne former qu'un seul buffer, les données DataL sont rangées au début du buffer et les données DataR sont rangées à la fin du buffer. Cette configuration permet l'utilisation de 2 bus **S88-N**, un à gauche et un à droite ayant la même taille et limité à 256 capteurs maximum chacun. A la demande du PC qui contrôle le réseau, celui-ci lit jusqu'à 10 fois par seconde les paquets de données recueillies sur le bus **S88** par l'Arduino qui lui transmet en moins de 15ms, sans que l'Arduino MEGA ne soit ni perturbé ni ralenti dans son fonctionnement. Cependant à chaque changement d'état des capteurs, les dernières données lues sur le bus S88 sont envoyées au PC Maitre pour l'informer du changement.

Cette fonction utilise 5 pins de l'Arduino : **Reset, Ps/Load, Clock, DataL et DataR**. Voir la page rétro-signalisation.

*** **Attention** : l'utilisation du bus S88 avec d'autres logiciels que WDD6, TCOwifi, JMRI et CDM-Rail n'a pas été testée, ni mise au point. ***

Bon train !

Philippe Chavatte