

J'ai enfin succombé aux charmes du protocole MFX !!!

MFX est une marque déposée pas Märklin

<https://www.maerklin.de/de/produkte/produkt-informationen/modellbahn-steuerung/mfx>

Cette technologie impose des décodeurs spécifiques qui possèdent un **identifiant unique** (UID) propre à chaque locomotive.

Elle passe par une phase d'apprentissage dans laquelle la Mobile Station va lire et mémoriser les principaux paramètres d'une locomotive.

Ce projet, démarré le 28 déc 2021, a été très enrichissant !!!

Il m'a permis de comprendre le dialogue entre ma centrale Mobile Station 2 et un décodeur sur la base de l'**excellent document produit par Stefan Krauss et son équipe** : <http://www.skrauss.de/modellbahn/Schienerformat.pdf>

J'ai d'abord voulu comparer les paquets produits par ma Mobile Station 2 et une locomotive avec ce document.

En réutilisant l'optocoupleur du §3 <http://gelit.ch/Train/DirectMM2.pdf>, j'ai longtemps cherché à acquérir les bonnes trames !!!

Même avec un code d'interruption minimaliste, il n'est pas possible d'extraire les trames (durée du bit = 100 µs) avec un Arduino Due cadencé à 84 MHz (et un suréchantillonnage de 10)

Heureusement, mon analyseur logique <https://www.lab-nation.com/> offre des extensions très intéressantes https://wiki.lab-nation.com/index.php/Creating_your_own_Protocol_Decoder qui m'ont permis d'observer les trames décrites par Stefan.

Fan du PWM implanté dans l'Arduino Due, j'ai pensé le réutiliser !?

A la réflexion, produire le signal MFX sans PWM avec digitalWrite est plus efficient ... et plus simple ; surtout pour les changements de polarité.

Je me suis un peu cassé les dents sur l'implémentation du CRC ; voir ligne 316 de <http://gelit.ch/Train/DirectMFX.ino>

Heureusement que internet regorge de trésors comme le **gigantesque développement de Daniel Sigg** : <https://sigsoftware.ch/wordpress/category/modellbahn/>

Mes vifs remerciements vont à Stefan et Daniel pour les précieux conseils qu'ils m'ont donnés !!!

Mon code Arduino Due mis à disposition se veut minimaliste et exige la présence de la Gleisbox pour lire l'UID mémorisé dans le décodeur et pour l'activer en mode MFX si la locomotive fonctionnait auparavant en MM2

Il peut permettre à un hobbyist de démarrer simplement avec :

- 1 aiguillage Adresse=5 en ligne 80
- 2 locomotives dont les UIDs doivent être modifiés (voir page suivante)
- et 16 rétrosignalisations

Les rétrosignalisations S88 (lignes 359 à 376) sont optionnelles

Elles sont lues chaque seconde (ligne 214) alors que mon code personnel lit 96 rétro chaque 20 ms

Le résultat est affiché avec la commande s (=Statistiques)

Les pins 10 à 13 sont utilisées (lignes 13 à 16) :

- `const int CLOCK = 10;`
- `const int DATA = 11;`
- `const int RESET = 12;`
- `const int LOAD = 13;`

Au niveau matériel, ne pas oublier d'ajouter le composant L293 à la carte Arduino Due

Voir §10 de <http://gelit.ch/Train/DirectMM2.pdf>

Les pins 2 et 3 de Arduino Due sont utilisées :

- `const int Pin2_L293 = 2;`
- `const int Pin7_L293 = 3;`

La pins 8 active ou non le signal électrique sur le rail :

- `const int PowerPin = 8;`

La pins 9 est optionnelle et renseigne sur l'état (PowerOFF ou PowerON) du rail :

- `const int Red_Led = 9;`

Des tests intensifs sur plusieurs semaines ont permis de valider ce code avec ma version qui gère 15 locomotives compatibles MFX.

J'en ai profité pour remplacer le décodeur du modèle 29486 qui ne supporte pas MFX

<https://www.maerklin.de/en/products/details/article/29486>

Mon choix s'est porté sur le décodeur ESU LokPilot 5 → <https://www.esu.eu/en/products/lokpilot/lokpilot-5/>

Son UID se trouve en ligne 77.

A propos de <http://gelit.ch/Train/DirectMFX.ino> :

- Ne pas oublier d'ajouter la librairie DueTimer dans Arduino IDE (Tools – Manage Libraries...)
- Utilisation de machines d'états (SM = State Machine) pour Interrupt et loop
- Dans **loop** (lignes 186-218), la séquence S=2 à S=5 respecte les timings (delay) en émettant les trames MFX après PowerON ; l'utilisateur active PowerON avec Enter.

Le compteur Za utilisé par Zentrale() est incrémenté après chaque getSID mais n'est pas mémorisé lors du PowerOFF car tous les décodeurs acceptent de toujours démarrer avec Za=1.

getSID alloue une adresse dynamique de 7 bit à l'UID correspondant.

Ces UID (lignes 76-77) doivent être modifiés pour correspondre aux locomotives utilisées

Voir procédure dans §4.1 de http://gelit.ch/Train/Raildue_F.pdf

L'état S=5 impose l'envoi de Zentrale toutes les 500 ms et Periodic toutes les 50 ms

Des conflits sur la mémoire tampon B[] sont donc possibles et gérés par Busy

La procédure Periodic est indispensable au bon fonctionnement ; elle envoie à toutes les locomotives enregistrées les valeurs de vitesse (133), direction (132) et les fonctions F0 à F15 (lignes 137-138)

- La procédure Func (112) est très utile car elle permet d'activer une seule fonction parmi 128 possibles (F0 à F127)
- Speed (ligne 100) n'utilise que 3 bits pour les vitesses Speed(0) à Speed(7)
- **L'interruption** (243-307) est exécuté chaque 50 µs (MFX) ou chaque 104 µs (Turn en mode MM2) ; le signal électrique généré par la carte Arduino présente 3 cas :

Emission continue du caractère de synchronisation (SM=10 en ligne 282)

Maintient le synchronisme des décodeurs, fournit l'énergie pour le mouvement, l'éclairage et le son des locs

Emission d'une trame MFX (SM=20 SM=30)

Longueur = variable Len

Envoi d'une commande MM2 destiné à un Turn (SM=3 à 8)

Pause=6.25 ms + 18 bit à 104 µs + Pause=1.5 ms + 18 bit à 104 µs + Pause=6.18 ms = 17.7 ms

Il m'a semblé utile de connaître l'occupation due aux trames MFX sur le rail

Mesuré avec un interval de 1 seconde, les trames MFX occupent un maximum de 40% à l'initialisation (getSID) et ne dépassent pas 20% avec mes 15 trains en mouvement simultanément

L'occupation due aux Turns n'est pas significative car chaque commande subit une pause = 250 ms (ligne 228)

- Les **statistiques** (commande s) affichent :

Les valeurs des variables

Power=1 S=5 SM=10

Détail de la dernière trame MFX envoyée

Len=48 CRC=93

1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0
0 1 0 0 1 1

Détail de la dernière trame MM2 envoyée

MM2= 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0

Détail des 16 bits de rétrosignalisation

S88 1=1 2=1 3=1 4=0 5=0 6=0 7=0 8=0 9=0 10=0 11=0 12=0 13=0 14=0 15=0 16=0