05 le réseau miniature

Le réseau miniature est constitué de différents éléments qu'il faut savoir commander dans un projet d'automatisation : aiguilles, cantons, signalisation lumineuse, passage à niveau, gare cachée, etc. Ce chapitre présente plusieurs solutions pour faire interagir les éléments entre eux afin de constituer un ensemble cohérent.

5.1 / ARCHITECTURE DES SOLUTIONS ÉLECTRONIQUES DANS UN RÉSEAU MINIATURE

Plusieurs solutions existent pour commander un réseau de trains miniatures avec une carte à microcontrôleur. Nous allons les passer en revue en insistant sur les avantages et les inconvénients.

Réseau miniature publié dans Loco-Revue (couverture de LR930)

RÉSEAU À CARTE UNIQUE DE GESTION

La première solution qui s'impose à l'esprit est d'avoir **une seule et unique carte pour gérer l'ensemble du réseau**. La **figure 5-1** montre ce genre d'architecture : on place des capteurs sur le réseau (détection d'occupation par exemple) et les signaux transmis sont mis en forme par une petite carte électronique (en vert clair) pour être compatibles avec Arduino. De la même façon, la carte Arduino envoie des signaux qui sont éventuellement amplifiés par une carte électronique (en jaune) pour les **actionneurs** (moteurs d'aiguilles, servomoteurs, etc.).

Si la carte s'occupe de tout, il lui faudra de nombreuses entrées-sorties pour être reliée aux différents capteurs et aux différents actionneurs (voir un peu plus loin). Il n'est pas forcément nécessaire d'avoir une carte qui travaille rapidement car nos modèles de trains ne se déplacent pas si vite que cela, en comparaison avec la vitesse de travail d'un microcontrôleur, même s'il n'a que 8 bits. **Une première limitation est donc le nombre d'entrées-sorties de la carte.**

La **figure 5-2** montre l'ensemble de tous les composants réunis autour d'une carte Arduino Mega 2560 pour le réseau **EX_MACHINA décrit dans Loco-Revue 936 de juillet 2025** (les poussoirs sont remplacés par des I.L.S sur le réseau). Comme on peut le voir, cette carte est déjà bien saturée et ce réseau est un petit réseau à deux aiguilles seulement. Si j'avais voulu mettre l'ensemble de la signalisation lumineuse, la carte n'aurait pas suffi et un choix sur les signaux a dû être fait. L'intégration des différentes fonctions a été possible car la carte Mega se contente de détecter les trains et de gérer le réseau comme un B.A.L, gérant à la fois la signalisation lumineuse et l'arrêt des trains devant un sémaphore. **Le programme a été écrit par une Intelligence Artificielle, ce qui a bien réduit le temps de développement.**

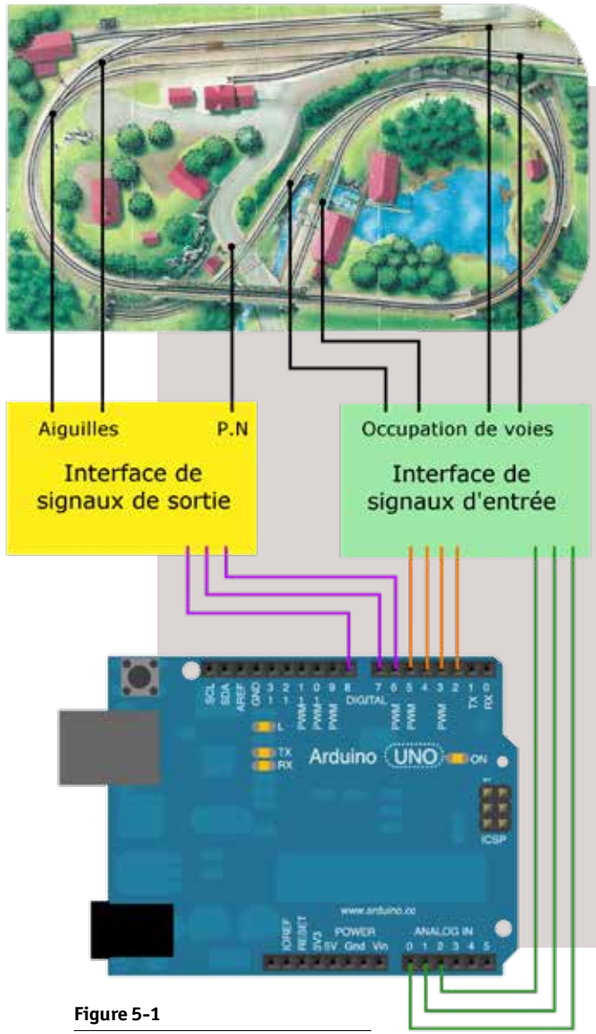

Figure 5-1

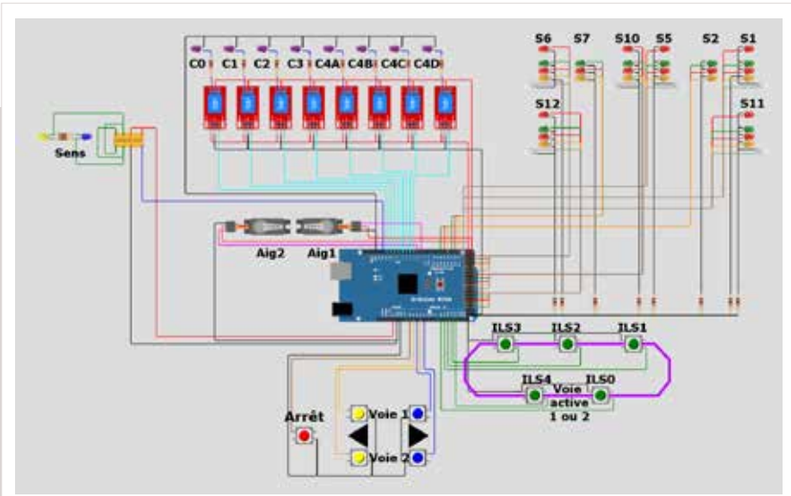

Figure 5-2

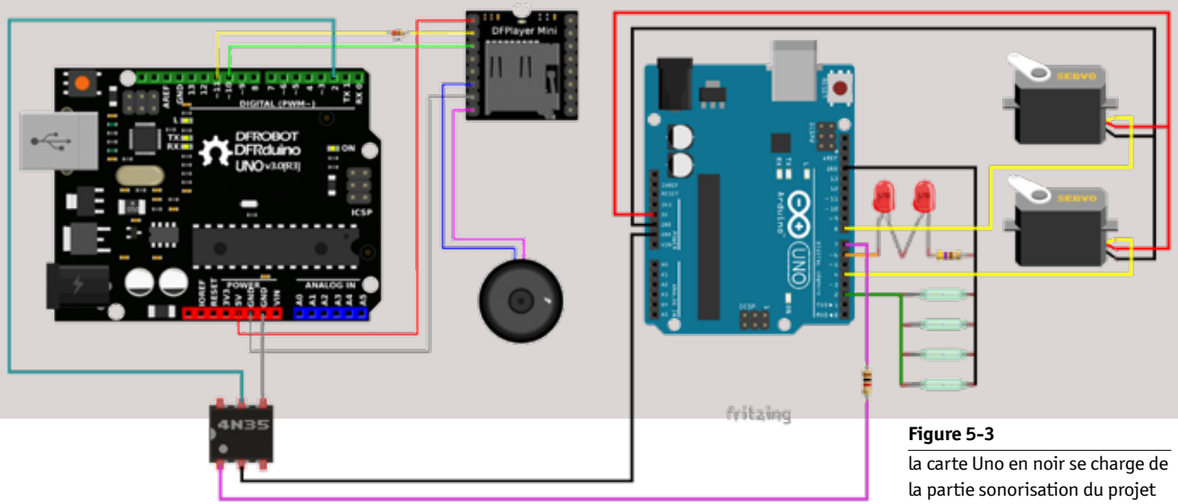

Figure 5-3
la carte Uno en noir se charge de la partie sonorisation du projet

Une deuxième limitation est la **difficulté à gérer du multitâches**. Ajouter une fonctionnalité dans un système déjà opérationnel, sans ajouter des erreurs qui peuvent se révéler critiques pour les autres fonctions, n'est pas toujours très simple. Par exemple, lorsque j'ai développé un PN (passage à niveau) pour le réseau Train In Box (**PN décrit dans Loco-Revue 894 de janvier 2022 et 895 de février 2022**),j'avais confié la partie sonorisation à une deuxième carte Uno (voir **https://locoduino.org/spip.php?article264**) car développer cela sur la carte qui gérait la détection des trains et le mouvement des barrières, créait des perturbations. J'aurais pu finir par trouver une solution mais certainement au prix de modifications drastiques de la partie non sonorisée du PN qui était parfaitement opérationnelle. Vu le prix d'une carte Uno, j'ai préféré en ajouter une deuxième qui ne fait que jouer le fichier sonore. La liaison entre les deux cartes est un simple fil : le niveau d'une sortie de la carte 1 commande, à travers un optocoupleur qui assure l'isolation galvanique, la carte 2 qui exécute le programme de lecture, comme le montre la **figure 5-3**.

Ceci nous amène à un autre concept : diviser pour mieux régner !

RÉSEAU À CARTES MULTIPLES SPÉCIALISÉES ET AUTONOMES

La **figure 5-4** montre un **réseau géré par plusieurs cartes Arduino** : chaque carte est spécialisée, la A et la B gèrent les aiguilles, la C ne gère que le passage à niveau et la D gère l'éclairage des bâtiments. Cette spécialisation fait que **les programmes n'interfèrent pas les uns avec les autres**, ce qui est plus simple à mettre au point. Le coût est bien évidemment plus élevé mais le temps gagné peut le justifier et **la maintenance du système est facilitée**. Il est parfois nécessaire que ce que fait une carte soit connu des autres cartes. Il faut alors les faire communiquer entre elles au moyen d'un bus de communication.

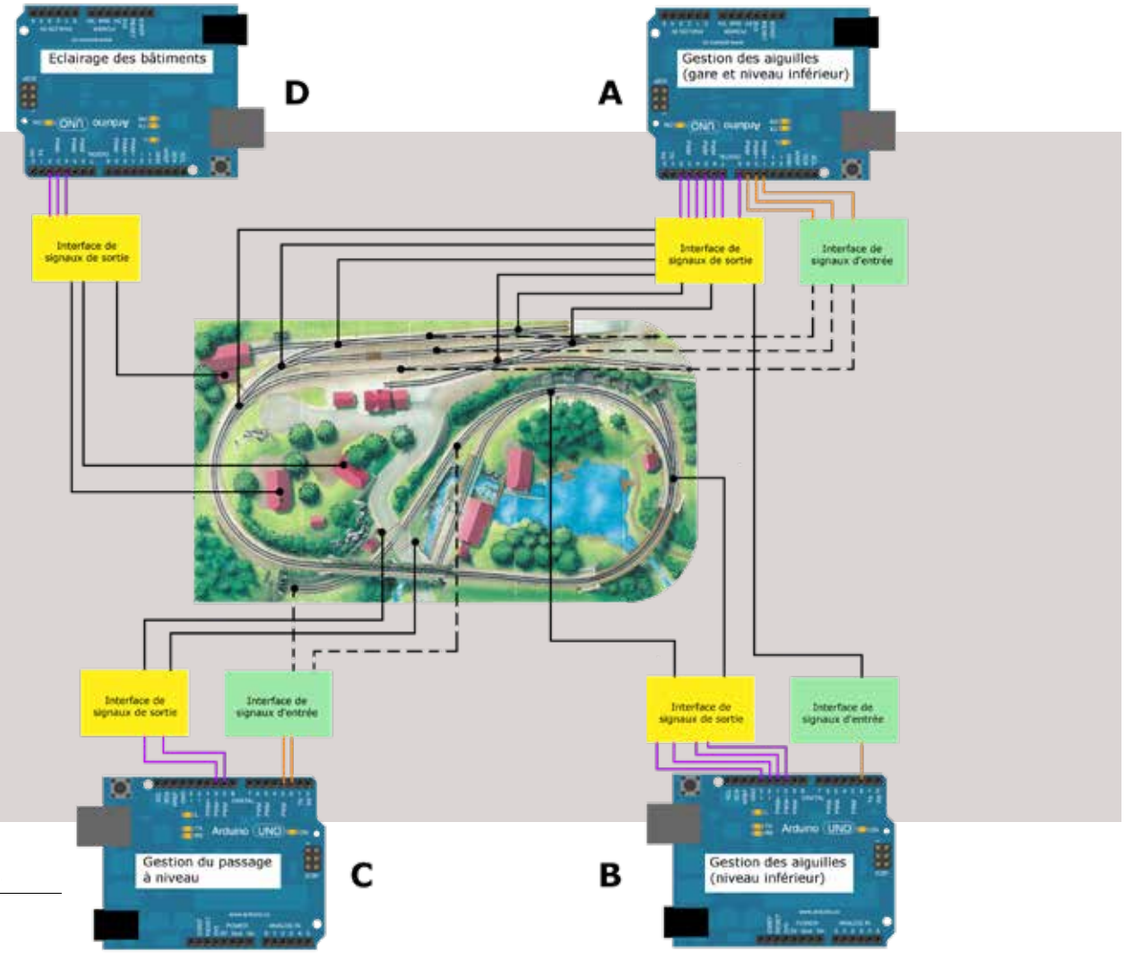

Figure 5-4

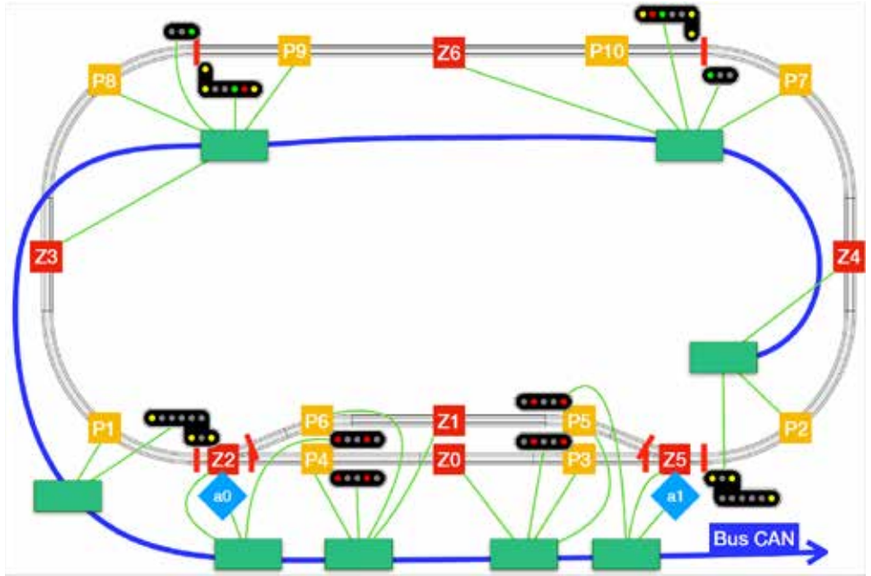

Figure 5-5
source LOCODUINO

RÉSEAU À CARTES MULTIPLES COMMUNIQUANT ENTRE ELLES

On peut imaginer que la gestion du réseau soit confiée à **plusieurs cartes à microcontrôleur communiquant entre elles**. Par exemple, une première carte peut détecter le train et communiquer à d'autres cartes sur quel canton il se trouve, quel est son itinéraire, s'il doit ou non s'arrêter en gare. Une deuxième carte, recevant les informations, peut alors se charger de la signalisation lumineuse, une troisième carte peut gérer les aiguilles pour former le bon itinéraire et une quatrième carte peut s'occuper du passage en gare et de l'éventuel arrêt.

Une telle architecture peut laisser penser qu'on est en train d'inventer une véritable usine à gaz, mais il n'en est rien. Au contraire, chaque carte est plus facile à mettre au point pour la fonction qui la concerne et **seule la communication reste le point principal à régler**. On peut aussi imaginer que les cartes ont toutes des fonctions identiques et sont identiques, ce qui fait descendre le coût de production, et qu'elles communiquent avec une carte gestionnaire : **ce principe a été introduit dans le modélisme ferroviaire en premier par LOCODUINO** qui a conçu des cartes satellites reliées à une carte gestionnaire du réseau. Chaque satellite pouvait assurer trois détections, permettre l'allumage de 9 LED et piloter un servomoteur. L'ensemble des satellites était relié **à un bus CAN (**voir plus loin) et recevait toutes les informations d'état du réseau et en tenait éventuellement compte pour agir sur le réseau. La **figure 5-5** montre ce concept. Les satellites sont les rectangles verts : ils sont tous reliés au bus CAN et sont connectés à des détecteurs de consommation de courant. Les signaux sont représentés, les détecteurs d'occupation de zone sont repérés par **Z**, les détecteurs de position par **P** et les servomoteurs d'aiguilles par **a**. Vous pouvez consulter l'ensemble de ce projet à cette adresse : **https://locoduino.org/spip.php?article242** .

DIFFÉRENTS BUS POSSIBLES POUR COMMUNIQUER

Plusieurs bus de communication font partie de l'écosystème Arduino et je vais décrire ceux qui sont les plus fréquemment utilisés pour communiquer avec un périphérique ou pour faire communiquer des cartes entre elles. Ce sont tous **des bus série**, ce qui veut dire que les bits qui constituent les octets des données à transmettre sont émis les uns après les autres, soit en commençant par le bit le moins significatif (bit 0), soit par le bit le plus significatif (bit 7).

SPI : l'acronyme signifie **Serial Peripheral Interface**. Ce bus est couramment utilisé pour interfacer des composants comme un **lecteur de tag RFID (Radio Frequency Identification)** ou bien des écrans LCD ou OLED. Il fait appel à la notion de contrôleur et de périphérique (anciennement appelée maître-esclave).

Le bus utilise quatre fils pour échanger les données entre contrôleur et périphérique :

- **Un fil SCK** avec les signaux d'horloge, permettant de synchroniser la transmission des bits
- **Un fil CIPO** (Controller In Peripheral Out) permettant l'échange dans le sens périphérique vers contrôleur (anciennement MISO master in slave out)
- **Un fil COPI** (Controller Out Peripheral In) permettant l'échange dans le sens contrôleur vers périphérique (anciennement MOSI master out slave in)
- **Un fil CS** (Chip Select) qui permet au contrôleur de sélectionner le périphérique avec lequel il veut communiquer (anciennement SS slave select)

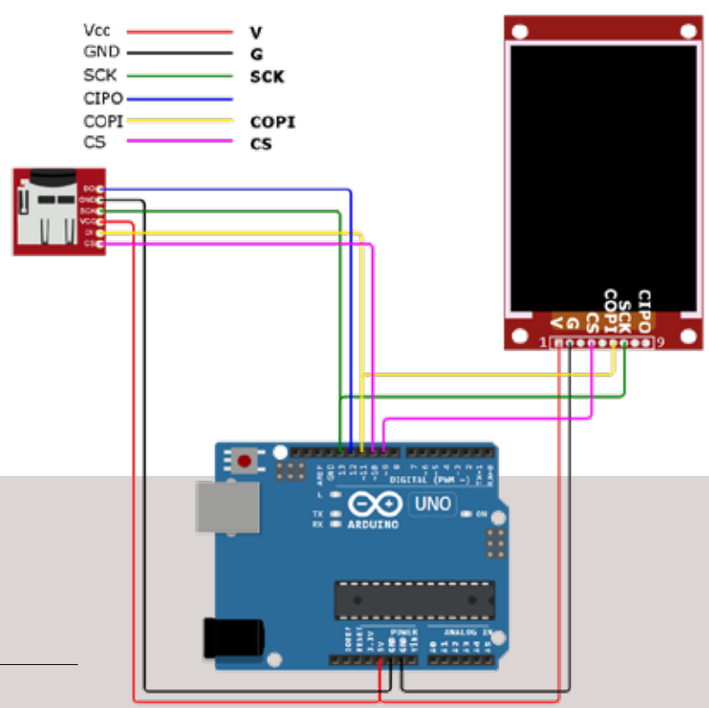

Figure 5-6

La **figure 5-6** montre une carte Arduino communiquant avec deux composants (un écran et un lecteur de carte SD) grâce au bus SPI pour lequel une bibliothèque existe. L'écran reçoit les données à afficher mais ne retourne rien : il n'y a donc pas de liaison CIPO.

Selon le mode de synchronisation entre horloge et donnée, le bus travaille selon quatre modes qui sont expliqués ici : **https://docs.arduino.cc/learn/communication/spi/** . Il faut aussi préciser la vitesse de transmission et si on commence par le bit le plus significatif (MSBFIRST) ou par le bit le moins significatif (LSBFIRST).

I2C : l'acronyme signifie **Inter-Integrated Circuits**. Le bus ne comporte que deux fils (plus le fil de masse) :

- **Le fil SCL** qui véhicule les signaux d'horloge
- **Le fil SDA** qui véhicule les données

La **figure 5-7** montre comment deux cartes peuvent être reliées en I2C pour échanger des données. Chaque périphérique a sa propre adresse (définie par construction) et peut ainsi recevoir la transmission qui le concerne. La bibliothèque Wire permet de définir soi-même ces adresses dans le cas de deux cartes.

L'utilisation du bus I2C est décrite ici : **https://docs.arduino.cc/learn/communication/wire/** Des exemples sont donnés dont on pourra s'inspirer pour un projet de modélisme ferroviaire.

CAN : l'acronymesignifie Controller Area Network. Simple, robuste et peu coûteux, le bus CAN est la solution idéale pour relier plusieurs composants sur un même réseau, et **LOCODUINO a été le premier à l'utiliser dans le cadre du modélisme ferroviaire**. C'est un bus différentiel, peu sensible aux parasites, constitué d'une simple paire de câbles torsadés CANHigh et CANLow. On trouve, pour un prix dérisoire, des petites cartes électroniques permettant d'interfacer des anciennes cartes Arduino (Uno, Nano, Mega) avec un bus CAN ; les nouvelles cartes sont souvent pourvues d'un contrôleur CAN mais elles nécessitent tout de même un transceiver (composant spécialisé). Il existe aujourd'hui plusieurs bibliothèques pour communiquer et mettre en œuvre un bus CAN.

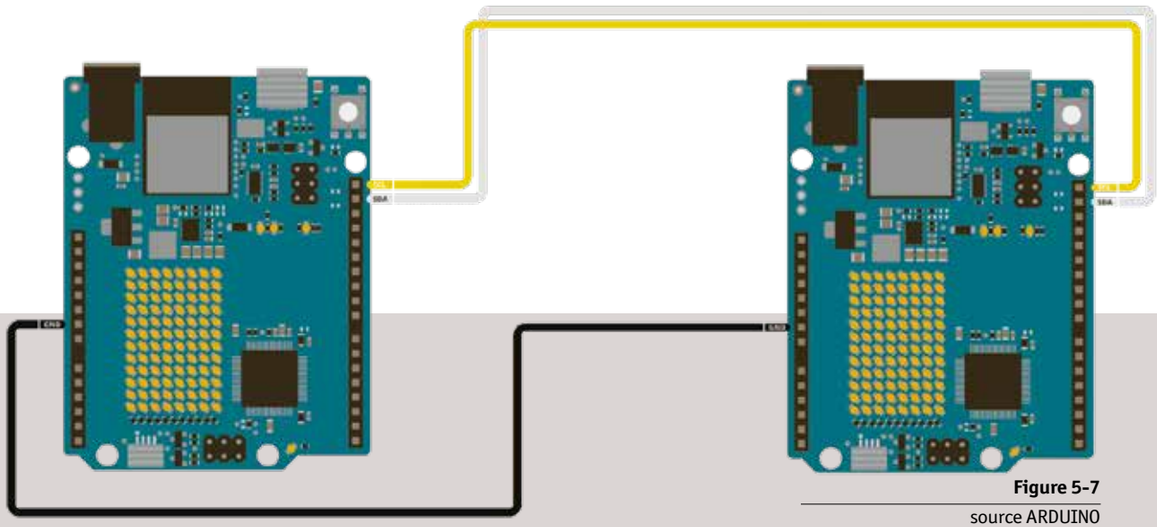

Figure 5-7
source ARDUINO

Cet article **https://locoduino.org/spip.php?article268** est parfait pour démarrer dans ce domaine, mais si vous voulez maîtriser ce bus, le livre **« CAN et CAN FD » de Pierre Molinaro**, chez Elektor, est vraiment la bible du CAN.

La **figure 5-8** montre comment relier deux cartes Uno en bus CAN : la carte de droite pourrait gérer un passage à niveau tandis que la carte de gauche s'occuperait de surveiller l'arrivée des trains en remplaçant les poussoirs par des I.L.S sur le réseau.

Ou bien encore, la carte de gauche pourrait être un TCO électronique envoyant ses ordres vers la carte de droite qui gèrerait les servomoteurs d'aiguilles et la signalisation lumineuse.

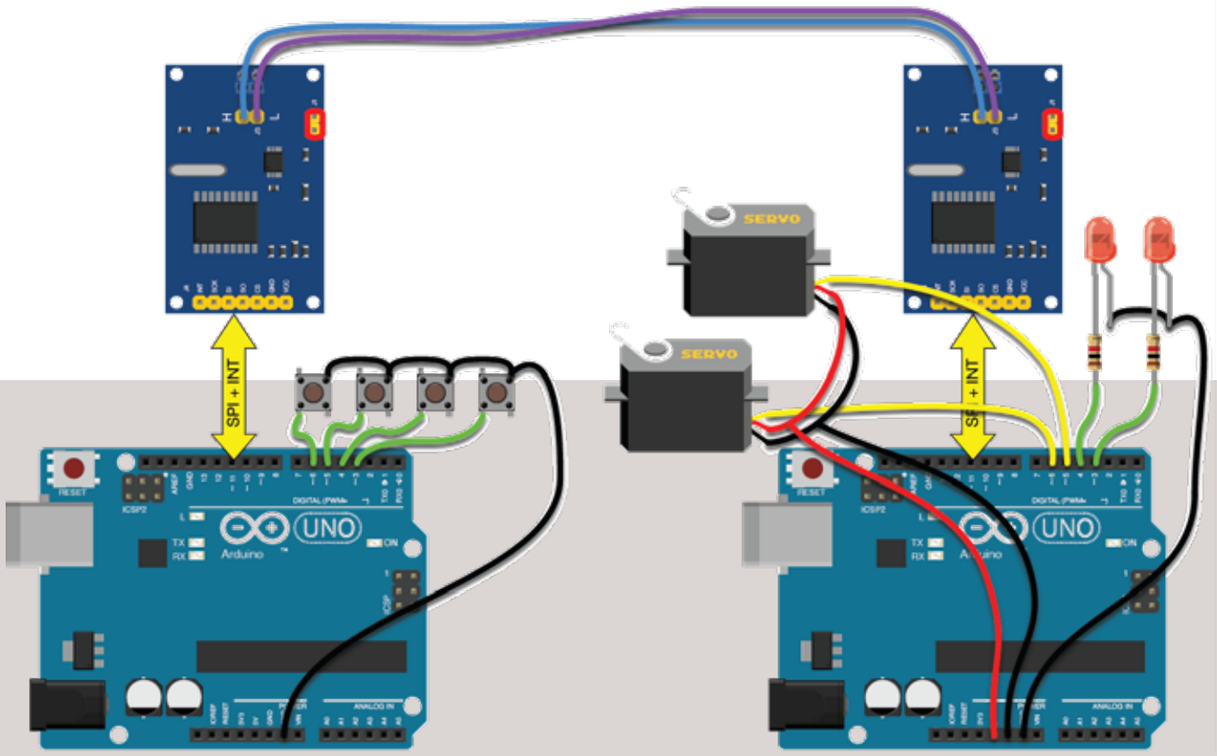

Figure 5-8
source LOCODUINO

5.2 / LES ACTIONNEURS DU MODÉLISME FERROVIAIRE

Dans ce paragraphe, je ne traiterai que des capteurs installés sur un réseau miniature. Les capteurs servent essentiellement à localiser les trains, mais une simple photorésistance peut aussi allumer l'éclairage du réseau lorsque la nuit commence à tomber. **Pour repérer les trains, nous avons deux solutions** :

- Le détecteur d'occupation
- Le détecteur de passage

Le **détecteur d'occupation** permet de savoir qu'un train occupe une portion de voie, sans savoir exactement où il se trouve sur cette voie. Il fournit un signal pendant une **durée relativement longue**, signal qu'il faut rendre compatible avec la tension possible sur l'entrée qui l'analyse (3,3 V ou 5 V). Comme le signal dure un certain temps (tant que le train se déplace sur la portion de voie), la carte ne peut pas le louper et le programme qui gère ce capteur est plus facile à écrire. Généralement, ce type de détecteur utilise la consommation de courant par la locomotive ou les wagons (**figure 5-9**) : **son comportement est donc meilleur en numérique car la voie reste alimentée en permanence**. En analogique, ce détecteur ne peut plus rien détecter lorsque la portion de voie n'est pas alimentée avec une locomotive à l'arrêt. Il faut faire appel à un autre système, généralement basé sur l'**injection sur la voie d'une tension ne servant qu'à la détection** et pas à la traction. Les détecteurs qu'on trouve dans le commerce sont assez onéreux, de l'ordre de 10 à 15 euros par zone où la détection doit se faire.

Le **détecteur de passage** permet de savoir qu'un train est à un endroit précis du réseau. Il peut être basé sur la détection d'un **champ magnétique** (I.L.S interrupteur à lames souples ou bien capteur à effet Hall), ou bien sur la détection ou l'interruption d'un **faisceau lumineux infra-rouge** (IR), ou encore sur un **lecteur RFID** (identification par radio fréquence). Si c'est un champ magnétique qui déclenche le détecteur, alors les locomotives doivent être équipées d'un aimant mais dans ce cas, seule la locomotive est détectée et par les wagons. Pour éviter cela, on peut aussi mettre un aimant sur chaque wagon : l'I.L.S ou le capteur Hall réagit alors à chaque véhicule, et locomotive et wagons peuvent être ainsi **comptés à l'entrée d'une zone et décomptés à la sortie**. Un détecteur d'entrée de zone peut aussi être considéré comme détecteur de sortie de la zone précédente car les zones se suivent (cas des cantons par exemple). En général, on se contente d'un aimant sur le wagon de queue, ce qui permet de contrôler qu'il n'y a pas de rupture d'attelage (voir chapitre 7).

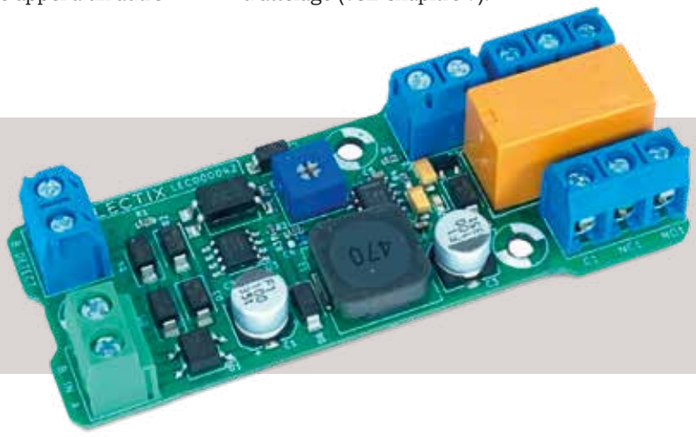

Figure 5-9

Ce genre de détecteur de passage est **facile à ajouter sur un réseau déjà terminé**, ce qui ne serait pas le cas des détecteurs d'occupation qui nécessitent un tronçonnage d'un rail et sa réalimentation en courant. L'I.L.S est le plus simple mais il faut avoir en tête que c'est avant tout un interrupteur mécanique et qu'il **produit des rebonds** que le logiciel devra savoir traiter. Le capteur à effet Hall n'a pas ce problème, mais il doit être alimenté en courant (généralement 5 V) ce qui fait un fil de plus. Le signal délivré par l'I.L.S ou le capteur Hall est **fugace** (le temps que l'aimant le survole) et le programme doit donc **surveiller le capteur en permanence** pour ne pas louper d'événements. Il est donc plus difficile à écrire et il faut souvent utiliser les interruptions externes du microcontrôleur.

Le **capteur RFID** est un capteur qui non seulement détecte le passage d'un train, mais **en plus peut l'identifier grâce à son tag** ; la technique a été décrite dans le tome 1 et je n'y reviendrai pas. L'identification par le programme permet de décider du trajet du train, de la voie qu'il doit emprunter, s'il doit s'arrêter en gare on non, s'il doit klaxonner, etc. La **figure 5-10** montre deux lecteurs de badge RFID qui peuvent se glisser sous une voie et le tag qui peut se coller sous la locomotive.

En conclusion, les capteurs sur un réseau vont **envoyer une information à la carte Arduino** et cette information doit être traitée avant d'être utilisée. Par exemple, s'il ne faut pas louper de survols d'un I.L.S, il ne faut pas non plus en compter plusieurs à cause des rebonds.

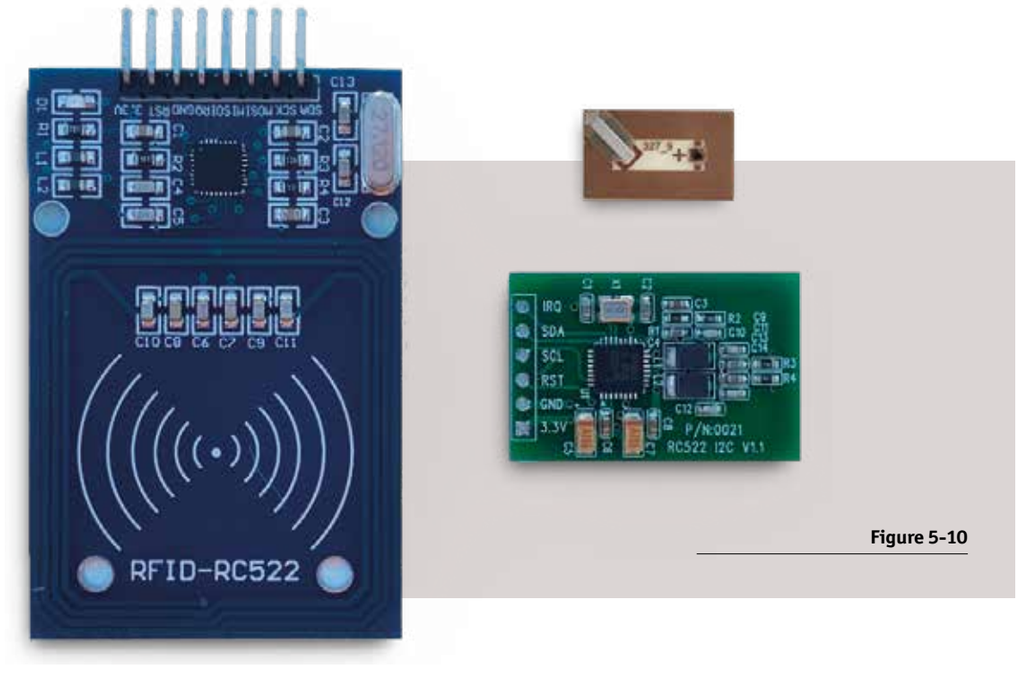

Figure 5-10

LES ACTIONNEURS DU MODÉLISME FERROVIAIRE

Dans ce paragraphe, je ne traiterai que des actionneurs installés sur un réseau miniature. En fonction de son programme, la carte Arduino **envoie des signaux vers des actionneurs pour produire un certain effet**, signaux qu'il faut parfois amplifier. Une simple **LED** est déjà un actionneur, par exemple celle qui équipe les feux de PN et c'est bien la carte qui la fait clignoter quand un train approche. Un **relais** alimentant une portion de voie est aussi un actionneur. Les **moteurs d'aiguilles** sont également des actionneurs, qu'ils utilisent un **servomoteur** ou bien un **moteur lent**, mais autant un seul servomoteur peut utiliser le 5 V fourni par la carte, autant le moteur lent nécessite sa propre alimentation.

Les moteurs électriques, qu'ils soient à **courant continu** (DC) ou bien **pas à pas**, sont également des actionneurs très prisés sur un réseau. Les premiers nécessitent une interface qui permettra de déterminer sens et vitesse de rotation. Certains circuits intégrés sont prévus pour cela mais on trouve aussi des cartes qui font le travail. Ces mêmes cartes peuvent aussi commander un moteur pas à pas bipolaire, alors que les moteurs pas à pas unipolaires nécessitent des cartes à base de circuit intégré matrice de transistor comme l'ULN2003 ou l'ULN2803. Tous ces moteurs permettent de **mouvoir les éléments de décor** du réseau comme un rotor d'hélicoptère, une roue de moulin, une grande roue de fête foraine, etc.

Enfin, les **écrans** peuvent servir à reproduire les espaces publicitaires d'un quai de gare de notre époque moderne ou bien les affichages municipaux d'un village, ce qui donne de très jolies animations. La mise en œuvre de ces actionneurs a été décrite dans le tome 1.

En conclusion, les actionneurs sur un réseau vont **recevoir des informations de la part de la carte Arduino**, les signaux étant le plus souvent amplifiés ou bien confiés à un bus de communication (voir plus haut).

LES CIRCUITS INTÉGRÉS INDISPENSABLES EN MODÉLISME FERROVIAIRE

Il existe deux sortes de **circuits intégrés (CI)**, les « traversant » (DIP pour **Dual Inline Package**) qui ont des broches qui traversent le circuit imprimé et se soudent côté verso, et les « CMS » qui sont des Composants Montés en Surface (SMD en anglais, pour **Surface MountedDevice**) dont les broches sont soudées côté recto du circuit imprimé. Les CMS sont beaucoup plus petits et donc plus difficiles à souder, même si on y arrive avec un peu de soin et d'entraînement. Lorsqu'on fait réaliser un circuit imprimé sur mesure, on peut demander à ce que certains CMS soient soudés dessus lors de la phase de fabrication. Aussi, les CI que vous devez avoir en stock sont plutôt des CI DIP qui peuvent s'enficher sur une breadboard (voir tome 1) afin de fabriquer un prototype du montage pour l'expérimenter. La **figure 5-11** montre la différence entre DIP et SMD : pour les DIP, il est toujours possible d'utiliser des supports de CI.

Je vais donc parler des **CI vraiment indispensables**, ceux qu'on trouve fréquemment sur les montages à carte Arduino. Je ne parlerai pas des composants discrets comme les diodes ou les transistors. Bien évidemment, la liste que je vous propose n'est pas exhaustive.

Les **optocoupleurs** sont fréquemment utilisés pour assurer l'isolation galvanique entre carte et périphérique dont les tensions d'alimentation sont différentes (3,3 V et 5 V par exemple ou bien 5 V et 12 V d'un réseau).

Les **matrices de transistors** comme l'ULN2003 ou l'ULN2803 sont utilisées pour amplifier les signaux de sorties et aussi servir d'inverseur de signal, ce qui peut permettre de brancher des signaux lumineux à anodes communes sur un montage réalisé pour des signaux à cathodes communes. Ces CI sont aussi utilisés comme interface pour moteur pas à pas unipolaire.

Les **CI pont en H** comme le L293D ou le L298 sont utilisés pour régler sens et vitesse de rotation de deux moteurs à courant continu ou bien encore comme interface pour un moteur pas à pas bipolaire.

Le **CI 74HC495** est un registre à décalage qui permet de commander 8 LED avec seulement trois sorties d'Arduino ; on l'utilise beaucoup pour la signalisation lumineuse du réseau quand le nombre de sorties de la carte se révèle insuffisant.

On peut ajouter à cette gamme différentes **portes logiques (soit NOR soit NAND)** ainsi qu'un **amplificateur opérationnel** comme le 741. Si vous avez besoin de bricoler une tension régulée, la gamme des **CI 78XX** (XX étant la tension positive désirée) et **79XX** (XX étant la tension négative désirée) est très pratique. Enfin, le **CI NE555** est considéré par les électroniciens comme le couteau suisse pour réaliser des montages, mais en électronique programmable, vous ne l'utiliserez pas souvent car le microcontrôleur peut reconstituer tout ce qu'il fait.

Ces différents CI étant fabriqués en grande série, ils sont bon marché et en avoir quelques-uns d'avance ne coûte pas une fortune et peut rendre service. Votre stock sera complété par des LED de différentes couleurs et une boîte de résistances de différentes valeurs, des diodes, des transistors (ceux à effet de champ sont plus souvent utilisés avec les cartes Arduino), des I.L.S, des poussoirs, des interrupteurs et quelques potentiomètres. Tout ce qui a un intérêt pour les montages Arduino se trouve **dans les kits d'initiation** qui sont souvent moins chers que les composants achetés séparément ; c'est pourquoi je ne fournis aucune valeur ou référence pour ces composants discrets.

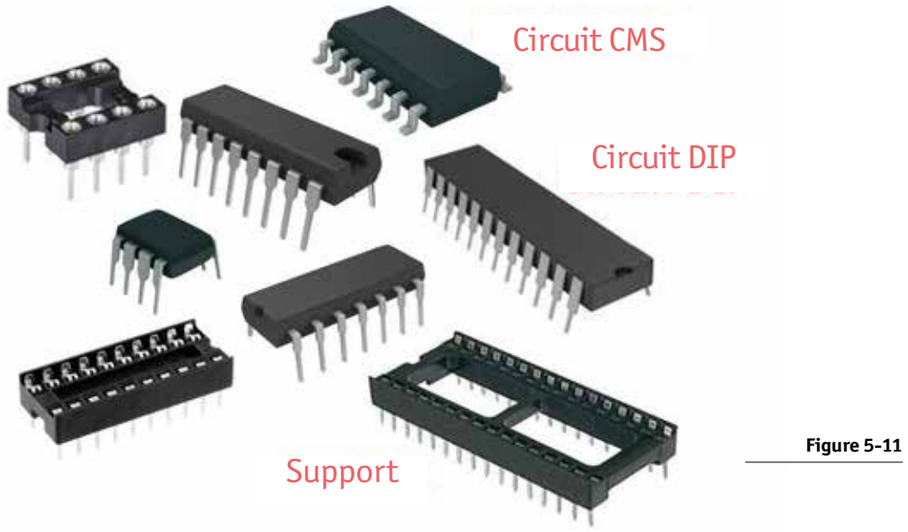

Figure 5-11

Utilisation d'écrans à Miniatur Wunderland à Hambourg, devant une foule enthousiaste

RÉSUMÉ DU CHAPÎTRE 5

Si vous avez acheté ce guide pour automatiser votre réseau, alors ce chapitre est important pour faire les bons choix lors de la conception d'un projet. Ce chapitre a rappelé qu'il y a plusieurs solutions d'architecture avec chacune des avantages mais aussi des inconvénients. Il faut donc bien réfléchir à toutes les possibilités car revenir en arrière plus tard est extrêmement difficile. Ce chapitre a aussi montré qu'on peut trouver dans le commerce de nombreuses cartes électroniques qu'il n'y a qu'à assembler entre elles pour réaliser les interfaces électroniques dont on a besoin.