Chapitre 14 - Vers une électronique moderne et simplifiée

Nous avons appris à construire différents modules capables de générer des signaux numériques périodiques ou non, et de les envoyer vers d'autres modules dans le but de les mettre en forme, de les compter, de les comparer, de les amplifier… Reste maintenant à mettre tout cela en pratique pour concevoir nos propres animations sur notre réseau.

Ainsi, par exemple, pour concevoir une horloge (générateur de signaux carrés périodiques), nous pouvons utiliser des triggers de Schmitt, des portes NOR ou NAND ou bien notre bon vieux copain le 555. Il faudra tout de même calculer les composants qui vont autour (et les trouver) pour que notre horloge batte la seconde. Vue l'incertitude sur la valeur des composants courants, est-ce que le résultat sera précis ? En modélisme, on n'est pas trop regardant…

Mais tout de même, s'il pouvait exister un composant à qui il suffirait de dire :

- allume la LED
- attends une demi-seconde
- éteint la LED
- attend une demi-seconde
- recommence indéfiniment

Ce serait vraiment beaucoup plus simple…

Et bien ce composant existe bel et bien et ne fait plus partie de la science-fiction ; il s'agit du **microprocesseur** et du **microcontrôleur**.

Le microprocesseur n'est absolument pas adapté à notre hobby car trop compliqué à mettre en œuvre. Et il a fallu des années avant que le microcontrôleur ne soit accessible à tous. Cette complexité a fait tourner les talons à de nombreux électroniciens amateurs et vous pouvez aussi, si cela vous parait compliqué, tourner les talons. Ou bien, par curiosité, rester et découvrir ces composants qui révolutionnent aujourd'hui le monde de la bidouille, du DIY (Do It Yourself ou faites-le vous-même), le monde des Makers (fabricants amateurs par opposition au monde industriel). Nous allons découvrir deux composants électroniques au même titre que ceux qu'on a déjà étudiés. Faites-moi confiance et restez, cela ne mord absolument pas.

Le microprocesseur

Il n'est pas question de décrire ici la mise en œuvre d'un microprocesseur, cela nous entrainerait trop loin. Nous allons simplement décrire ce composant et voir pourquoi on ne l'utilise pas (ou très peu) en modélisme ferroviaire.

Un microprocesseur est un composant capable de réaliser des **opérations arithmétiques et logiques**. Vous connaissez des opérations arithmétiques comme l'addition de deux nombres ou leur soustraction. Vous connaissez aussi la multiplication et la division. Mais au fait, savez-vous qu'une multiplication, ce n'est rien d'autre qu'une série d'additions ? En effet, 3 x 4, c'est 3 + 3 + 3 + 3, quatre fois. Et la division ? Rien d'autre qu'une série de … soustractions. Par exemple, j'ai quatorze billes que je veux partager entre Michel, Gérard, Alain et Jean-Claude. Je commence à prendre 3 billes que je soustrais de mon compte pour les donner à Michel ; il m'en reste 11. Je fais la même opération pour Gérard qui reçoit également 3 billes. Idem pour Alain et Jean-Claude. Que me reste-t-il ? 14 - 3 -3 -3 - 3 = 2. Dommage, ce n'est pas assez pour refaire une tournée ! 14 (billes) divisé (ou partagé) par 4 (copains), c'est 3 chacun et il en reste 2.

Notre microprocesseur sait faire des additions et des soustractions donc il sait faire les quatre opérations. Mais une opération logique ? Vous savez aussi ce que c'est puisqu'on a fait de la logique dans le chapitre 9. Ce sont les opérations AND, OR, NOT, etc.

Un microprocesseur possède une **unité arithmétique et logique** (en anglais ALU) avec laquelle il peut faire ces opérations. Cela ne suffit pas. Le microprocesseur a besoin **de mémoire de deux types** : la mémoire ROM (qui ne peut qu'être lue) et la RAM qui peut être lue ou écrite. Dans la ROM, le microprocesseur va chercher les ordres, c'est-à-dire son programme (par exemple partager les billes). Dans la RAM, il ira chercher les données (nombre de billes, nombre de copains). Et pour faire tout cela, il lui faut aussi un métronome qui lui donne le rythme ; le microprocesseur a un **circuit d'horloge** en lui qui doit simplement être ajusté avec un cristal de quartz qu'on ajoute à l'extérieur du boitier (pour que le rythme soit le plus précis possible). Lorsqu'il a terminé le calcul qu'on lui demande, le microprocesseur va ranger son résultat dans la mémoire RAM, puisqu'il peut écrire dedans.

Et ensuite, comment fait-on pour connaitre le résultat ? On se miniaturise pour aller voir les électrons ?

Le microprocesseur a besoin de **périphériques pour communiquer avec le monde extérieur**. Parfois très simple comme quelques LED ou un afficheur sept segments pour les sorties, un clavier à 16 touches pour les entrées, le périphérique peut être plus compliqué comme un écran ou un clavier ou un disque dur. Mais pour que la communication se fasse entre le microprocesseur et le périphérique, il faut **un circuit d'interface** : appelons-le PIA (Peripheral Interface Adapter), bien il y ait d'autres noms possibles.

Et les signaux dans tout cela ? Ils transitent d'un composant à l'autre **via des bus**, c'est-à-dire un ensemble de lignes de signaux ayant une fonction bien précise. Un microprocesseur a en général besoin de trois bus : un **bus d'adresse** qui lui permet d'aller dans la bonne case mémoire chercher ce qu'il veut, un **bus de données** qui achemine jusqu'à lui la donnée qu'il est allé chercher et un **bus de contrôle** par où transitent les différents signaux pour contrôler le bon fonctionnement du microprocesseur.

J'ai essayé de rester simple et j'ai forcément fait des raccourcis ; par exemple, la RAM ne contient pas que des données car elle peut aussi contenir des programmes, ceux que vous écrivez vous-même. Et la ROM peut aussi contenir des données comme des constantes nécessaires au bon démarrage de l'ensemble du système conçu autour du microprocesseur. Les données sont organisées en octets (ensemble de 8 bits pouvant avoir chacun la valeur 0 ou 1), ce qui fait que le bus de données contient 8 lignes par lesquelles transitent les signaux. Enfin, la RAM s'efface dès que l'alimentation électrique est coupée ; à la mise en route, le microprocesseur exécute le programme inscrit dans la ROM qui ne s'efface pas hors tension.

La figure 14.1 montre la complexité d'un système à microprocesseur et on comprend vite pourquoi ce composant n'est pas adapté à notre hobby ; en fait, le microprocesseur est idéal pour construire un micro-ordinateur car celui-ci a besoin de beaucoup de mémoire, dispose de périphériques sophistiqués (clavier, écran, lecteur de disque dur ou optique) et il y a de la place pour tout cela. L'avantage du microprocesseur est qu'il est rapide pour travailler et qu'il peut gérer de grandes quantités de mémoire (plusieurs Giga-octets soit plusieurs milliards de cases mémoires pouvant accueillir chacune un octet).

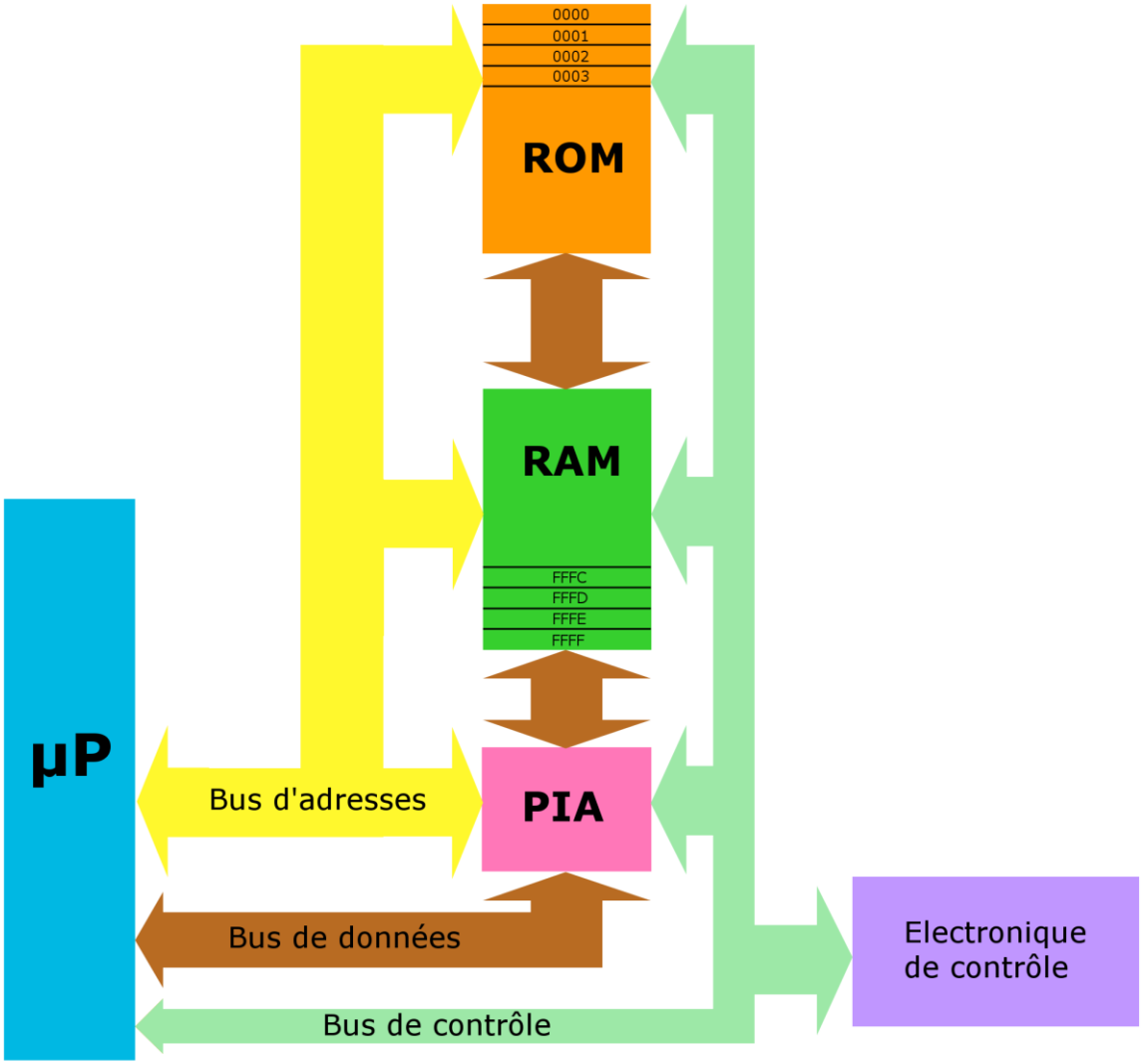

Figure 14. 1

Les premiers microprocesseurs étaient des microprocesseurs 8 bits (aujourd'hui, on en est à 64 bits) et ils s'appelaient 8080, 8085, 6800, 6502, Z80, F8, SC/MP. Ils possédaient chacun un jeu d'instructions (les ordres élémentaires qu'on peut leur faire exécuter pour réaliser les opérations arithmétiques et logiques). Mais la principale difficulté du microprocesseur, c'est qu'il faut écrire le programme à partir de ce jeu d'instructions et que ce n'est pas simple ! Et encore moins simple à dépanner quand ce programme ne fonctionne pas !

Donc, définitivement, oublions le microprocesseur dans un réseau de trains miniatures (encore que certains ont entièrement automatisé leur réseau à partir de carte architecturée autour d'un microprocesseur).

Le microcontrôleur

Imaginez qu'on puisse faire tenir toute la figure 14.1 dans le boitier d'un circuit intégré…

Bien sûr, nous n'allons pas avoir des milliards de cases mémoire pour stocker le programme ou les données, mais juste ce qui est nécessaire pour faire tourner un petit programme : quelques milliers d'octets de ROM et quelques centaines d'octets de RAM. Ce composant, c'est le **microcontrôleur**, une sorte de micro-ordinateur dans un boitier de CI.

Le composant doit permettre à chacun de réaliser ses propres applications ; il est donc hors de question de le programmer en usine, chacun doit pouvoir le faire avec un petit programmateur. De même, il n'est pas question d'écrire un programme en octets ; c'est très indigeste. Mais les constructeurs vont très vite mettre à disposition de tous, des petits logiciels capables de transformer un programme écrit dans un langage symbolique (l'assembleur) en langage machine (une suite d'octets) que le microcontrôleur peut comprendre.

Et avec cela, le microcontrôleur est devenu le circuit intégré universel ?

Que nenni ! D'abord, parce que les premiers programmateurs coûtaient assez cher et que l'assembleur, ce n'est pas évident. Cela ressemble à cela :

```
;     Sous-programme tempo
;
tempo     MOVLW     0xFF          ; on met 255 dans le registre W
          MOVWF     nombre1       ; nombre1 est initialisé à 255
          MOVWF     nombre2       ; nombre2 est initialisé à 255
tempo1    DECFSZ    nombre1, F    ; décrémentation et saut une fois à zéro
          GOTO      tempo1        ; nombre1 non nul
          MOVLW     0xFF          ; on remet 255 dans le registre W
          MOVWF     nombre1       ; on recharge nombre1 à 255
          DECFSZ    nombre2, F    ; décrémentation et saut une fois à zéro
          GOTO      tempo1        ; nombre2 non nul
          RETURN                  ; fin du sous-programme, nombre2 nul
```

Si vous y comprenez quelque chose, vous avez bien de la chance, car moi, je n'y comprends plus rien ou presque (et pourtant, c'est moi qui ai écrit ce bout de programme quelques années auparavant !).

L'arrivée des microcontrôleurs PIC (de Microchip) va pourtant aider les bricoleurs, d'autant que cette arrivée coïncide avec l'arrivée d'internet dans les foyers. On trouve alors tout un tas d'idées sur la toile pour programmer les PIC avec les moyens du bord et il devient possible de fabriquer de petits programmateurs reliés à un micro-ordinateur qui prend en charge l'essentiel de la tâche. Mais l'assembleur reste l'assembleur !

L'avantage du microcontrôleur sur le microprocesseur, c'est qu'il n'y a rien à rajouter sauf peut-être le cristal de quartz et encore, certains microcontrôleurs savent s'en passer au prix d'une légère imprécision souvent pas bien grave. Son inconvénient, il gère moins de mémoire, il est moins rapide, il est aussi moins puissant. S'il n'y avait pas ce fichu assembleur, ce serait tout à fait le composant idéal pour nos petits trains électriques…

Les modules programmables à base de microcontrôleur

Depuis quelques années, on voit apparaitre de petits modules conçus autour d'un microcontrôleur d'un prix très bon marché. Ces modules comportent tout ce qu'il faut pour faire fonctionner le microcontrôleur (de quoi l'alimenter en courant, le quartz pour son horloge et des connecteurs pour récupérer des signaux). Mieux, il n'y a qu'à relier le module à un ordinateur via un câble USB pour l'alimenter en courant et pouvoir le programmer. Ces modules portent le nom d'Arduino ou de Raspberry et il y en a (ou en aura) sûrement d'autres très prochainement.

Mais c'est toujours un microcontrôleur ? Au paragraphe précédent, nous nous étions arrêtés à la difficulté d'avoir un programmateur (acheté chèrement ou bricolé durement) et à la difficulté du langage assembleur !

Avec les modules, ces deux points sont résolus :

Le programmateur : c'est votre ordinateur qui va envoyer les signaux de programmation au microcontrôleur du module via le câble USB.

L'assembleur : il est remplacé par un langage évolué, un peu comme de l'anglais. Certains pensent qu'il faut apprendre le C à la place de l'assembleur, ce qui ne serait pas plus facile. En fait, vous pouvez programmer le module en C si vous connaissez ce langage, mais si vous ne le connaissez pas, il suffit d'apprendre moins d'une cinquantaine d'ordres pour pouvoir programmer le module. Ces ordres sont traduits en langage machine par votre ordinateur. Quant au programme qui fait cela, on le trouve gratuitement en téléchargement libre sur le site du constructeur qui met également à votre disposition toute la documentation nécessaire.

Quand je vous disais qu'il serait agréable d'avoir à dire :

- allume la LED
- attends une demi-seconde
- éteint la LED
- attend une demi-seconde
- recommence indéfiniment

Pour un module Arduino, cela correspond à :

- digitalWrite (LED, HIGH) ;
- delay (500) ;
- digitalWrite (LED, LOW) ;
- delay (500) ;

Il vous a fallu apprendre **deux mots nouveaux** (digitalWrite et delay) pour réaliser ce multivibrateur astable de fréquence 1 Hz !

Et le « recommence indéfiniment » ? Il le fera tout seul !

Utiliser un microprocesseur ou un microcontrôleur, c'est faire de **l'électronique programmable**, par opposition à **l'électronique câblée** à base de composants. Nous allons voir maintenant pourquoi c'est plus simple et moins cher.

L'électronique programmable

Encore une fois, ce chapitre va se contenter de survoler le sujet des modules de type Arduino (ou Raspberry) ; si vous êtes intéressés pour vous initier à ce nouveau genre d'électronique, vous pouvez consulter la suite de ce cours qui y sera consacrée dans le même état d'esprit que la première partie que vous avez déjà lue (vous apprendre à réaliser de petites applications très simples).

Mais revenons à nos moutons. Dans ce cours, je vous ai appris à réaliser des fonctions simples pour produire des signaux et les traiter comme il se doit pour qu'ils produisent une action sur votre réseau (arrêter un train, allumer une maison, etc.). En général, un montage en électronique câblée est constitué de plusieurs étages : l'alimentation, l'horloge, l'étage de comptage, l'étage amplificateur. C'est ce dernier étage qui amplifie le signal pour actionner par exemple un relais.

Avec un module de type Arduino, on remplace les différents étages d'un montage par le module lui-même **et surtout par son programme**, ce qui signifie qu'on remplace des composants électroniques par du programme informatique. Plus besoin de calculer des valeurs de composants, ni de souder ces composants ; en plus, si on se trompe, il suffit de réécrire quelques lignes de programme ou de corriger la valeur d'une variable dans le programme. On réalise donc des économies et on gagne du temps par rapport à un montage avec circuit imprimé et composants soudés.

Les seuls étages électroniques qui restent sont les interfaces entre le module et le réseau pour que le réseau envoie des signaux propres et adaptés au module et pour amplifier les signaux qui partent du module vers le réseau. De toute façon, il aurait aussi fallu concevoir et réaliser ces interfaces en électronique câblée pour relier réseau et montage.

On voit que les modules de type Arduino font gagner du temps et de l'argent, à condition de savoir programmer mais cela aussi, ça s'apprend. Le langage de base qu'on trouve sur le site du constructeur est composé d'une soixantaine de fonctions et d'une dizaine de structures de contrôle, mais on n'a pas besoin de tout connaitre au début. Avec très peu d'apprentissage et un esprit logique, on y arrive très vite.

Un autre avantage de l'électronique programmable est la souplesse qu'elle communique à un montage ; **il suffit de changer le programme pour obtenir un comportement différent** du module, alors que le montage du module avec les composants du réseau est le même et n'est pas modifié. Dans l'exemple ci-dessus, quatre lignes de programme ont suffi à fabriquer un multivibrateur astable ; si je change maintenant le programme, je peux obtenir un monostable ou un bistable.

Comme je vous l'ai dit, certaines fonctions sont difficiles à produire en électronique câblée alors que c'est très facile en électronique programmable. C'est par exemple le cas pour la temporisation du chauffard dans le montage des feux tricolores routiers ; en électronique programmable, il suffit de rajouter un ordre delay(2000) pour que le feu passe au vert deux secondes après que le feu de l'autre route soit passé au rouge (l'argument de delay, ici 2000, est exprimé en millisecondes et 2000 ms font 2 secondes).

Pour agir sur le réseau, les modules de type Arduino ont des connecteurs ; ils peuvent servir d'entrées ou de sortie (il suffit de les programmer comme tel). En entrée, le signal provient du réseau (un ILS par exemple), en sortie, le signal part du module pour aller actionner quelque chose sur le réseau (un relais pour interrompre le courant sur la voie par exemple). Les modules sont de plus capables de traiter des signaux analogiques (la tension prélevée aux bornes d'une LDR) ou numériques (signal de fermeture d'un ILS).

Ce que vous pouvez faire avec des modules de type Arduino ne dépend que de votre imagination.

Conclusion

Je comprends parfaitement que ce genre d'électronique puisse vous faire peur car il y a quelques années, elle n'était pas à la portée de tout le monde. Aujourd'hui, avec les progrès technologiques, il est devenu très facile de programmer des microcontrôleurs. À vous de voir si cela vous plait et si vous vous sentez prêt à apprendre cette nouvelle technique. Pour ma part, cela fait quelques années que j'ai remplacé l'électronique câblée par l'électronique programmable, car cela me parait plus simple. Mais je dois avouer aussi que je me suis un peu remis à cette électronique câblée pour vous préparer ce cours et que c'était très agréable.

À retenir sur les composants programmables :

- Le microprocesseur est un composant non autonome qui nécessite autour de lui de nombreux autres composants comme de la mémoire ROM ou RAM et des interfaces pour communiquer avec des périphériques.
- Le microcontrôleur est un composant autonome qui nécessite que très peu d'autres composants et qui réunit dans un même boitier un microprocesseur et tout son environnement de travail.
- Le microcontrôleur est moins rapide et moins puissant qu'un microprocesseur mais il est mieux adapté pour de petites applications, notamment dans le cadre du modélisme ferroviaire.
- La difficulté d'utilisation des microcontrôleurs venait du fait qu'il fallait les programmer en langage assembleur avec un programmateur spécial.
- Cette difficulté est maintenant résolue par l'emploi de modules à base de microcontrôleur programmable avec un simple ordinateur en langage évolué et compréhensible.
- Ces modules permettent de remplacer l'électronique traditionnelle par de l'informatique pour traiter des mêmes fonctions de base, le programme pouvant être modifié autant de fois que nécessaire.
- Les modules présentent des connecteurs permettant de les relier avec le monde environnant, par exemple un réseau de trains miniatures.
- L'électronique programmable est beaucoup plus souple pour faire évoluer un montage sans avoir tout à recâbler.