Numérique et Sciences Informatiques
Chapitre IV - Composants intégrés d'un système sur puce
Travaux Dirigés 08

Classe de terminale

Dans ce TD, nous allons piloter des objets connectés : des LED (ou DEL) de couleur.
Plusieurs systèmes de codage de couleur existent. Nous allons utiliser le codage RVB (RGB en anglais).
Chaque pixel coloré d'une image encodée en 8 bits par couche est représentée par un triplet de nombres codés chacun sur 8 bits. Le premier nombre correspond à la composante rouge du pixel, le deuxième à la composante verte du pixel et le troisième à la composante bleue du pixel.

1. (a) Combien y a-t-il de valeurs différentes pour le codage de la couche rouge d'un pixel ?
   (b) En déduire le nombre de couleurs possibles pour un pixel.

Exemples de codage RVB

couleur	codage RVB
Rouge	(255,0,0)
Jaune	(255,255,0)
Vert	(0,255,0)
Cyan	(0,255,255)
Bleu	(0,0,255)
Magenta	(255,0,255)
Blanc	(255,255,255)
Gris	(50,50,50) ou (67,67,67) ou (220,220,220) ou ect.
Noir	(0,0,0)

2. (a) Lancer votre navigateur *Google Chrome* ou *Mozilla Firefox* puis naviguer vers le site MakeCode.
   (b) Créer un nouveau projet, donnez lui le nom « TD08 ». Choisissez l'option de code « Python seulement ».
   (c) Branchez la carte *Micro:bit* sur un port USB de l'ordinateur.
   (d) Ajouter l'extension « Neopixel ». Elle permet d'utiliser les Leds de couleur.

3. La carte de Leds est branchée sur la carte *Micro:bit* sur le port *pin0*. Elle comporte 24 Leds numérotées de 0 à 23.
   Faites glisser le code `Neopixel`▸`Neopixel sur broche ...` vers la partie de code. Le code s'écrit automatiquement. Il est modifiable. Une variable `strip` est alors créée. Elle correspond à l'ensemble des Leds. Ainsi :

   - `strip[0]` correspond à la *led1*
   - `strip[1]` corresponda à la *led2*
   - etc.

   Enfin, pour donner une couleur à une led, on attribue un tuple correspondant au code RVB à la variable correspondant à la led choisie. Par exemple, pour coloré la *led1* en rouge, on écrira :

```
1 couleur = neopixel.rgb(255, 0, 0)
2 strip.set_pixel_color(1, couleur)
```

   Il faut ensuite mettre à jour la carte *Halo* en utilisant la méthode `show()` de la classe `NeoPixel`.

```
1 strip.show()
```

   Dans la suite du TD, on pourra aussi utiliser la méthode `clear()` de la classe Neopixel afin d'éteindre toutes les leds.

4. Colorer la *led2* en vert.

5. Faire évoluer la *led3* du noir(éteinte) au bleu.

6. Effacer toutes les leds et allumer la *led1* en bleu.

7. On appellera led active la led allumée avant un événement. Ecrire un programme qui :

   - allume la led suivante (dans le sens trigonométrique) si on appuie sur le bouton B
   - allume la led précédente si on appuie sur le bouton A.

   Dans les deux cas, elle éteindra la led active.

8. Ecrire un nouveau programme qui allume toutes les leds avec des couleurs aléatoires et qui changent toutes les secondes.

9. Exporter le programme et le téléverser sur une carte *Micro:bit*.