



Electronics Project

Day 06 : RGB & advanced timers

contact@42chips.fr

Résumé: Musique électronique et vomi de licorne

Chapitre I

Préambule

Recette pour préparer de délicieuses chips triangulaires façon tortilla !

Dans un saladier, mélanger la farine de maïs, le sel, le paprika, le curry et le poivre. Choisir la quantité d'épices selon vos goûts.

Versez l'huile au centre puis mélanger un peu. Verser ensuite l'eau tiède puis mélanger la pâte avec les mains.

Former une boule puis laisser la pâte reposer environ 30 min à température ambiante. A la fin du temps de repos, préchauffer votre four à 170°.

Prendre un tapis silicone ou une feuille de papier cuisson de la taille de votre plaque de four. Fariner le dessus. Couper la pâte en 2 parties.

Étaler finement la première partie sur le tapis ou la feuille, saupoudrer de paprika puis repasser le rouleau sur la pâte.

A l'aide d'une roulette à pizza, former un rectangle puis faire des traits verticaux et horizontaux pour former des carrés.

Passer ensuite la roulette en biais pour obtenir des triangles.

Ne pas séparer la pâte et faire cuire tel quel, les tortillas vont se séparer facilement. Faire la même chose avec le restant de pâte.

Enfourner la première plaque pendant 15 et 20 min. Faire attention à la coloration de la pâte, cela peut aller vite à la fin du temps de cuisson.

Laisser refroidir à température ambiante, faire cuire la seconde fournée de tortillas et détacher les triangles de la 1ère fournée.

Les tortillas doivent être fines et croustillantes !

Chapitre II


Consignes générales

Sauf contradiction explicite, les consignes suivantes seront valables pour tous les TPs

- Le langage utilisé pour ce projet est le C.
- Il n'est pas nécessaire de coder à la norme de 42.
- Les exercices sont très précisément ordonnés du plus simple au plus complexe. En aucun cas nous ne prendrons en compte ni n'évaluerons un exercice complexe si un exercice plus simple n'est pas parfaitement réussi.
- Vos exercices seront évalués par des responsables de l'association 42Chips.
- Vous ne devez laisser aucun autre fichier que ceux explicitement spécifiés par les énoncés des exercices dans votre répertoire lors de la peer-évaluation.
- Toutes les réponses à vos questions techniques se trouvent dans les **datasheets** ou sur Internet. A vous d'utiliser et d'abuser de ces sujets pour comprendre comment réaliser votre exercice.
- Vous devez utiliser la datasheet du microcontrôleur qui vous est fourni et commenter les parties importantes de votre programme en renseignant où vous avez trouvé les indices dans le document, et, si nécessaire, expliquer votre démarche. Ne faites pas des pavés non plus. Il faut que cela reste clair.
- Vous avez une question ? Demandez à votre voisin de droite ou de gauche. Vous pouvez demander sur le salon dédié dans le discord de la piscine ou en dernier recours à un staff.

Chapitre III

Campbell's Soup I


	Exercice : 01
Rouge Vert Bleu	
Dossier de rendu : <i>ex01/</i>	
Fichiers à rendre : Makefile , main.c	
Fonctions Autorisées : avr/io.h , util/delay.h , avr/interrupt.h	

Vous devrez écrire un programme qui contrôle la LED RGB D5


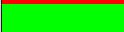
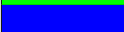




- la LED doit s'allumer en rouge, puis en vert, puis en bleu en boucle
- En changeant toutes les 1 seconde.

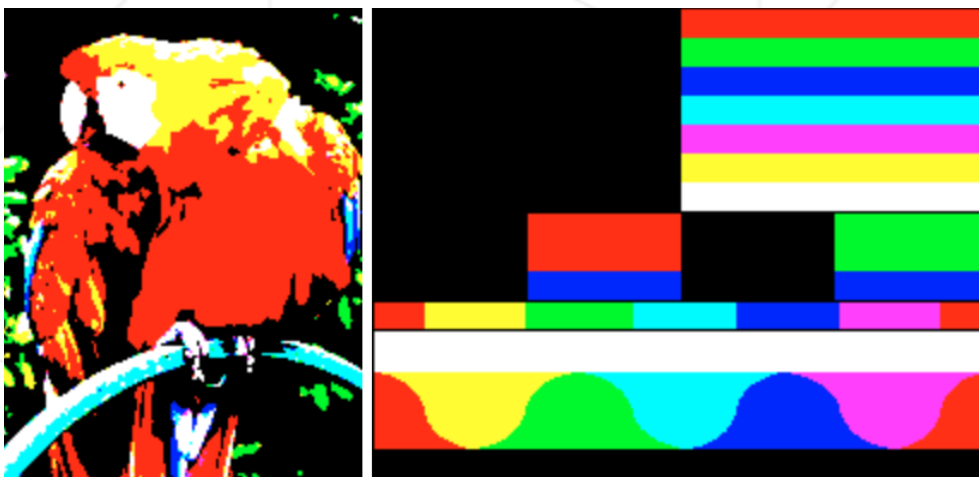
Chapitre IV


Technicolor

	Exercice : 01
3-bit RGB color	
Dossier de rendu : <i>ex01/</i>	
Fichiers à rendre : Makefile , main.c	
Fonctions Autorisées : avr/io.h , util/delay.h , avr/interrupt.h	

Ok maintenant plus de couleurs! Vous devrez écrire un programme qui contrôle la LED RGB D5 Voici un tableau des couleurs qui doivent s'afficher successivement. En boucle et en changeant de couleur toutes les secondes.

nom	R#	G#	B#	couleur
rouge	ff	0	0	
vert	0	ff	0	
bleu	0	0	ff	
yellow	ff	ff	0	
cyan	0	ff	ff	
magenta	ff	0	ff	
white	ff	ff	ff	



	Exercice : 02
24-bit RGB color	
Dossier de rendu : <i>ex02/</i>	
Fichiers à rendre : <i>Makefile, main.c</i>	
Fonctions Autorisées : <i>avr/io.h, util/delay.h, avr/interrupt.h</i>	

Ok maintenant encore plus de couleurs ! Vous devez écrire un programme qui contrôle la LED RGB D5 mais avec du PWN

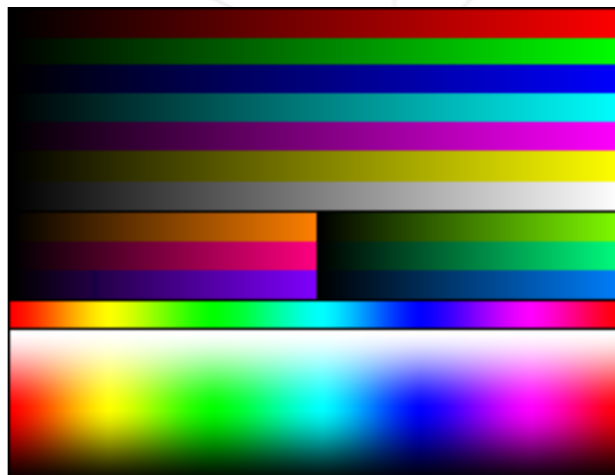
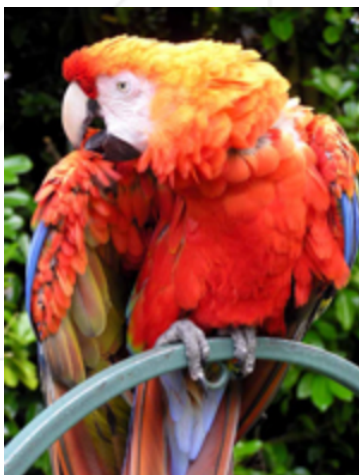
- Ecrivez une fonction `init_rgb()` qui initialise les timers.
- Puis écrivez une fonction `set_rgb()` qui permet de set la couleur de la LED.
- Pour finir votre programme doit afficher la roues des couleurs.



Les couleurs obtenues dans le gradient demandé sont toujours la somme de deux couleurs primaires, pas plus, parfois moins.

```
void set_rgb(uint8_t r, uint8_t g, uint8_t b);

void wheel(uint8_t pos) {
    pos = 255 - pos;
    if (pos < 85) {
        set_rgb(255 - pos * 3, 0, pos * 3);
    } else if (pos < 170) {
        pos = pos - 85;
        set_rgb(0, pos * 3, 255 - pos * 3);
    } else {
        pos = pos - 170;
        set_rgb(pos * 3, 255 - pos * 3, 0);
    }
}
```



Chapitre V

LM2512A



Exercice : 03

UART to RGB

Dossier de rendu : *ex03/*

Fichiers à rendre : **Makefile**, **main.c**

Fonctions Autorisées : **avr/io.h**, **util/delay.h**, **avr/interrupt.h**

Vous devez réaliser un programme qui écoute sur le port série. Qui quand on lui envoie une nouvelle ligne avec une couleur HEX RGB au format **#RRGGBB**, set la couleur sur la LED RGB D5



Il est normal que les couleurs ne s'affichent pas exactement comme vous le voulez. Il est possible de corriger le problème avec un **filtre logiciel**. Mais il ne vous est pas demandé le corriger.