www.didel.com//EduC-NotesFonctions.pdf

Edu-C – Notes techniques ou pédagogique à propos des fonctions EduC

Notes citées dans www.didel.com/educ/EduC-Fonctions.pdf

Note 01

DelMs();, comme **delai();** accepte aussi un délai déclaré comme byte. mais la valeur ne peut alors pas dépasser 256. Le C n'est (heureusement) pas trop rigoureux concernant les conversions de type pour les nombres entiers. Dans le cas des flottants c'est plus important.
DelMs(v16); le délai max est ~32000 si déclaré par int v16 (donc signé).

Note 02

Le concept de macros est important à bien expliquer ultérieurement, pour les différencier des fonctions. Elles permettent beaucoup. On ne les utilise ici que pour exprimer le comportement bas niveau du matériel.

Note 03 et 04

Le PWM ne s'applique pas seulement aux moteurs (pour lesquels le PFM est préférable).
Arduino a la fonction analogWrite, qui génère du PWM et suscite la confusion avec un signal vraiment analogique.
Pour une LED, la fréquence PWM doit être supérieure à 50 Hz.
La sensibilité de l'oeil est logarithmique. Une Led avec 1% de PWM est déjà très visible. Avec 10% de PWM, on voit une demi-intensité. Il y a naturellement des escaliers de perception puisque l'œil voit la différence entre 1 et 2% de PWM (mais pas entre 50 et 51%).
EduC offre les fonctions PwmLed-G-D-etc de 0 à 255, et les fonctions PwmG-D-etc de 0 à 31 seulement avec une réponse linéaire, en passant par une table de correction. Ainsi, un Pwm de 16 correspond à ~50% d'intensité. On peut facilement changer cette table. Des valeurs de 0 à 99 pourraient paraître préférables. La table serait plus grande et nos applications ne demandent pas plus de résolution. EduC cherche à montrer les contraintes matérielles et ce qui est naturel au processeur.
La valeur Pwm dans Rouge(), etc..est saturée à 32.

Note 05

Le problème des rebonds de contact est ignoré tant que l'on n'a pas les instructions pour l'analyser et bien l'expliquer. Je verrai volontiers une macro `DelR;` pour "délai rebond" pour remplacer le DelMS(20);

Note 06

En cherchant une valeur 8 bits précise, on voit qu'il est difficile de l'obtenir. Le circuit permet 10 bits (AnalogRead() d'Arduino) mais annoncer 16 bits pour ensuite réduire à 8 bits ou moins complique.

Note 07

Le HP se programme comme une LED. Un On ou Off fait entendre un clic.

Note 08

L'affichage 7-segments partage le port kidules et les 8 leds vertes. Un niveau logique 0 en sortie allume le segment. et éteint la diode.

Note 09

Le comportement de l'affichage, des LEDs et des signaux sur le connecteur Ki ne peut bien se comprendre que si on connait les circuits logiques. Il manquait une sortie dans le processeur AVR328 et il a fallu astucer pour éviter que les leds vertes distraient dans certaines tâches, et consomment du courant.. Le port Ki est câblé sur les mêmes sorties que l'affichage 7-segments, mais un transistor n'active les leds de sortie que si on a fait HpOn, donc la membrane du haut.parleur est attirée. Il ne fait pas de son, mais consomme. On ajoute donc la consommation des Leds, avec un maximum de 70mA à 3.7V, contre unb minimum de 10mA avec seulement le Oled.

Le Oled utilisé pour des textes et des dots ne pose pas de problèmes. Pour des jeux, c'est différent.

jdn 180214/180301