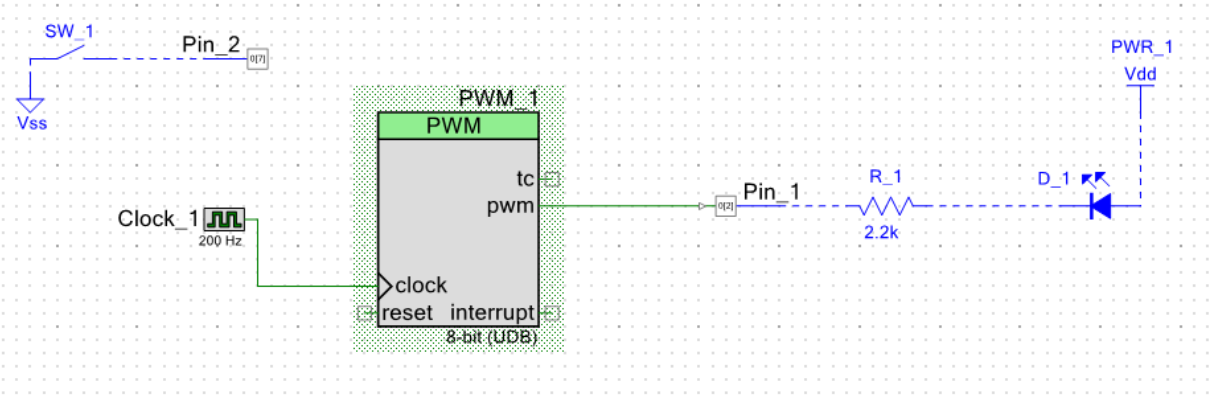
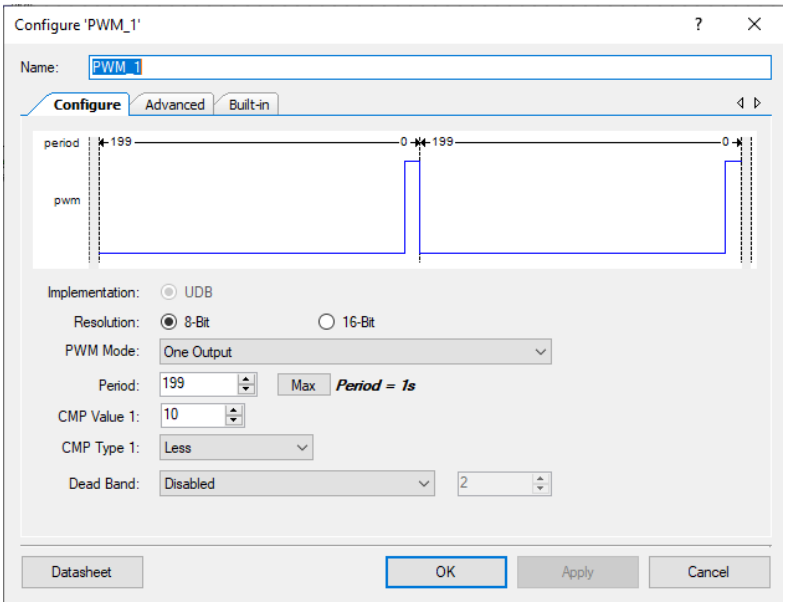
Programmation PSoC : utilisation de circuits avancés

Utilisation du PWM, « *Pulse Width Modulation* », pour le contrôle de LEDs

Réalisez le circuit suivant :



Vous configurerez le circuit de PWM de la manière suivante :



Le PWM est construit de la manière suivante :

- il démarre à partir d'une valeur choisie ;
- à chaque cycle de l'horloge dont la fréquence est choisie, il décrémente la valeur du compteur ;
- suivant une comparaison utilisant :
  - une valeur de référence choisie (appelée ici « *CMP Value 1* » ;
  - un opérateur de comparaison choisi (ici « less ») ;
  - la valeur courante du compteur ;
- la valeur en sortie du PWM est le résultat de cette comparaison (sur la figure, la valeur est vraie pour les valeurs 199 à 11, puis fausse pour les valeurs 10 et 0) ;

Attention

Pour déclencher le circuit de PWM vous devrez appeler la fonction C :

```
void PWM_1_Start(void);
```

Questions :

1. Sachant que la fréquence de l'horloge est de 200Hz, quelle est la durée de la période ?
   Combien de temps :
   - ◇ prends la décrémentation d'une valeur du compteur ?
   - ◇ dure l'allumage de la LED ?

   Quelle le pourcentage du temps d'allumage de la LED par rapport au temps d'extinction ?
   Est-ce que l'on peut s'en servir pour donner l'impression que la LED est moins lumineuse ?

2. Créez le circuit demandé en utilisant le bouton pour :
   - ◇ activer le clignotement de la LED lorsque le bouton est appuyé ;
   - ◇ éteindre la LED si le bouton n'est pas appuyé ;

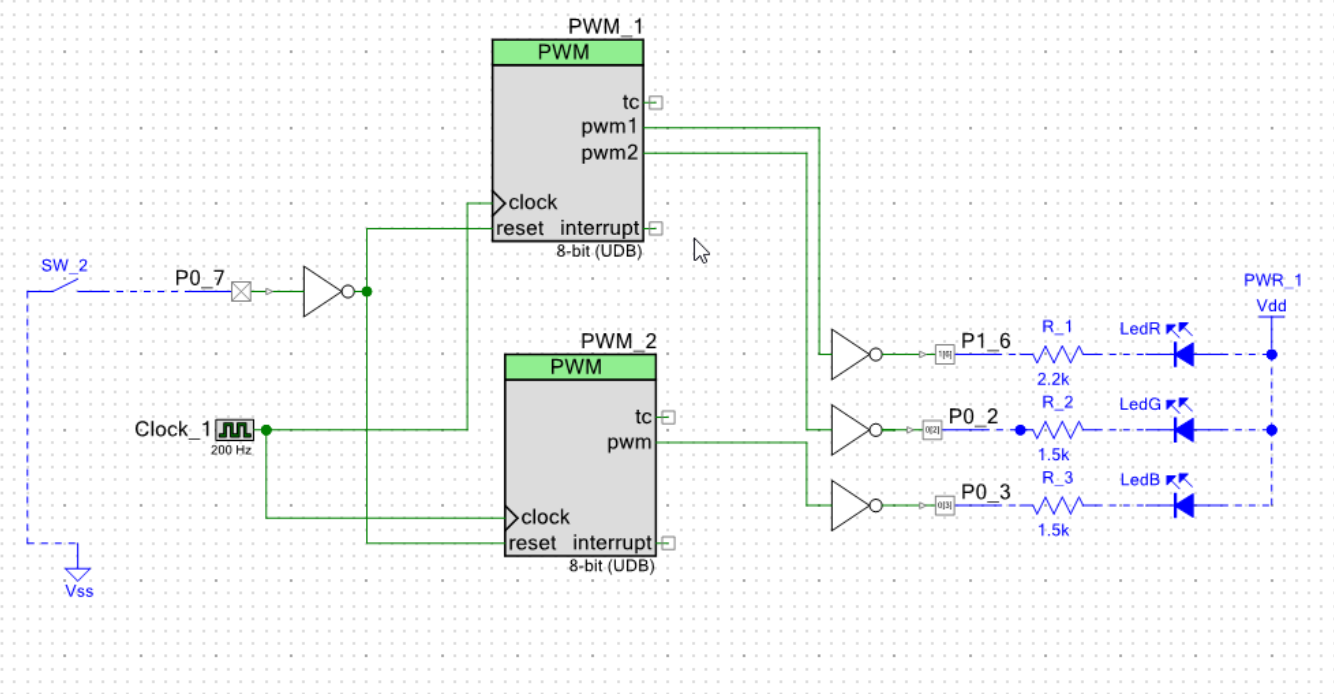
   Faut-il utiliser un programme C ou un circuit logique pour faire ce travail ?

3. Écrivez un programme C permettant lors de l'appui du bouton de passer d'un temps d'allumage de 25ms au lieu de 250ms.

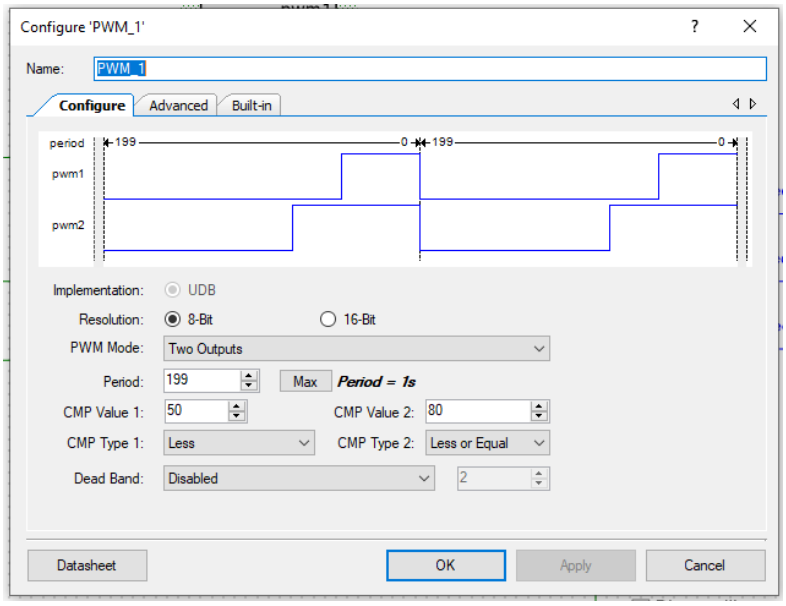
   Les fonctions C à votre disposition sont :

`void PWM_1_WriteCompare(uint8 valeur)`	*change la valeur utilisée dans le comparateur*
`uint8 Pin_1_Read()`	*retourne la valeur lue sur la broche au moment de l'appel*

Réalisez le circuit suivant :



Vous configurerez le PWM de la manière suivante :



3. Quel est le rôle du bouton ?
   À quoi sert le NON logique sur le bouton ? Sur les LEDs ?
   Que se passe-t-il si on maintient le bouton appuyé ? Est-ce normal ?

4. Écrivez un programme modifiant les 3 PWMs de manière aléatoires toutes les secondes
   Les fonctions C à votre disposition sont :

`void PWM_1_WriteCompare(uint8 valeur)`	*change la valeur utilisée dans le comparateur*
`void PWM_2_WriteCompare(uint8 valeur)`	*change la valeur utilisée dans le comparateur*
`uint8 P0_7_Read()`	*retourne la valeur lue sur la broche au moment de l'appel*
`void CyDelay(uint32 milliseconds)`	*délai de retour de la fonction exprimé en millisecondes*
`int rand()`	*retourne une valeur aléatoire. Vous pouvez utiliser* `rand()%256` *pour une valeur aléatoire entre 0 et 255*