Formation STI2D SIN :
« Chaîne d'information et Microcontrôleur PSOC »

Objectifs de la formation :

- Découvrir les microcontrôleurs PSoC,
- Faire lien entre le microcontrôleur PSoC et la représentation de la chaine fonctionnelle,
- S'initier à leur programmation avec le logiciel « PSoC Creator » et le kit associé,
- Mettre en œuvre les PSoC dans des projets.

Compétences abordées : « SIN2 - Valider des solutions techniques »

- CO2.1. rielle me
- CO2.2. me
- CO2.3. Traduire sous forme graphique l'architecture de la chai ne d'information i tres d'utilisation du simulateur
- CO2.4. me pour valider le choix d'une solution.

STI2D PADIOLLEAU Jean-Luc

Système d'information et numérique 8 novembre 2013

http://sti.ac-orleans-tours.fr

Contenu

1 Présentation générale des microcontrôleurs « PSOC »

1.1 Les différentes familles de microcontrôleurs « PSOC »

Le microcontrôleur PSOC, pour **p**rogrammable **s**ystem **o**n **c**hip, est un composant de traitement numérique programmable. Il est développé par la société américaine CYPRESS depuis l'année 2000. La particularité de cette puce électronique est qu'elle intègre, autour d'un cœur de processeur, des structures logiques et analogiques programmables, configurables et interconnectables.

Plusieurs familles se sont succédées :

- La famille « PSOC 1 », première à être développée, construite autour d'un processeur M8C.
- La famille « PSOC 3 » construite autour d'un processeur 8051.
- La famille « PSOC 5 » construite autour d'un processeur ARM Cortex M3.

Voici un tableau résumant les performances de chaque famille :

	PSOC1	PSOC3	PSOC5
Vitesse	24 MHz pour 4 MIPS	67 MHz pour 33 MIPS	67 MHz pour 84 MIPS
Taille du bus de données	8 bits	8 bits	32 bits
Mémoire programme	Flash de 4 ko à 32 ko.	Flash de 8 ko à 64 ko	Flash de 32 ko à 256 ko
Mémoire de donnée	SRAM de 256 o à 2 ko	SRAM de 2 ko à 8 ko	SRAM de 16 ko à 64 ko
Alimentation	1,7 V à 5,25 V.	0,5 V à 5,5 V	de 2,7 V à 5,5 V
Consommation	Actif : 2 mA Veille : 3 µA	Actif : 0,8mA Veille : 1 µA Hibernation : 200 nA	Actif : 6 mA hibernation : 300 nA
Conversion A/N	1 Delta-Sigma de 14 bits	1 Delta-Sigma de 20 bits	1 Delta-Sigma de 20 bits 2 SAR de 12 bits
Conversion N/A	2 de 8 bits	jusqu'à 4 de 8 bits	jusqu'à 4 de 8 bits
Communication	USB 2.0, I2C, SPI, UART, LIN	USB 2.0, I2C, SPI, UART, CAN, LIN, I2S, JTAG	USB 2.0, I2C, SPI, UART, LIN, I2S
Entrées / sorties	64	72	70

1.2 Logiciels de développement pour « PSOC »

Pour développer des applications autour des microcontrôleurs PSOC, 2 logiciels IDE[(1)] sont fournis par CYPRESS :

- **PSOC Designer** pour la famille PSOC 1, est basé sur la réalisation d'interconnexions matricielles entre différents blocs existants représentant les fonctions analogiques et logiques intégrées dans le microcontrôleur. La programmation séquentielle est définie en langage « C » grâce au compilateur intégré.

- **PSOC Creator** pour les familles PSOC 3 et 5, est basé sur un logiciel de saisie de schéma permettant de relier les symboles quasi-normalisés des fonctions analogiques et logiques intégrées dans le microcontrôleur. La programmation séquentielle est assurée par un compilateur « C ».

*(1) **IDE** : Integred Development Environment = Environnement de développement intégré, il s'agit d'une suite logicielle*

1.3 Kits d'évaluation pour « PSOC »

Il existe plusieurs kits permettant de s'initier aux microcontrôleurx PSoC selon la famille souhaitée et les fonctions abordées :

Référence	*Famille PSOC*	*éléments périphériques*
CY8CKIT-001	1, 3 et 5	Boutons poussoirs, LCD, LEDs, Capsenses, RS232, USB, WIRELESS (nécessite extension), potentiomètre, zone de prototypage "LABDEC"
CY8CKIT-050	5	Boutons poussoirs, LCD, LEDs, Capsenses, RS232, USB, potentiomètre, zone de prototypage à souder
CY3210-PSoCEval1	1	Boutons poussoirs, LCD, LEDs, RS232, potentiomètre, zone de prototypage à souder

CY3210-PSoCEval1

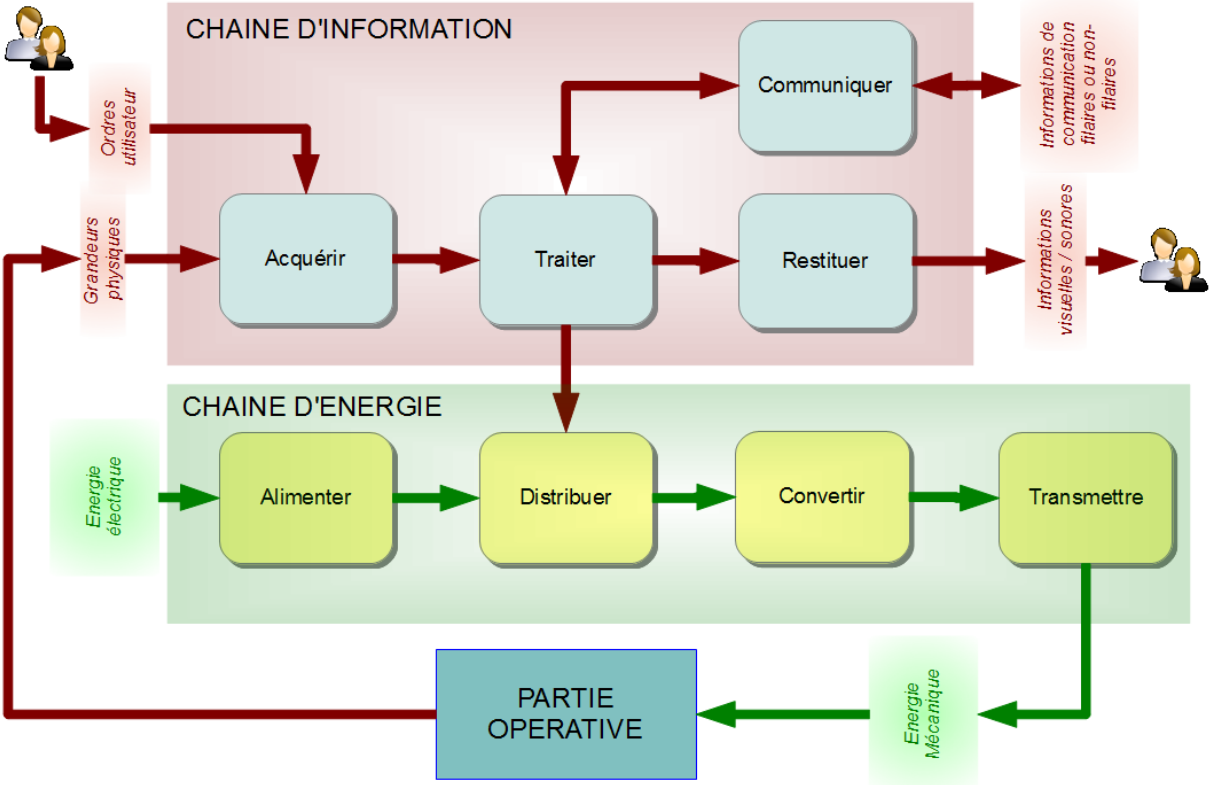
CY8CKIT-001

CY8CKIT-050

2 Le PSOC dans la chaine fonctionnelle

2.1 Schéma fonctionnel

La représentation fonctionnelle graphique des systèmes mécatroniques répond au schéma suivant :



On distingue 2 parties : la chaîne **d'information** et la chaîne **d'énergie**. Les microcontrôleurs « PSoC » s'intègrent parfaitement dans la chaîne d'information.

2.2 Chaîne d'information

Fonction	*Constituants*	*Eléments du « PSOC 5 » s'y rapportant*
Acquérir	Capteurs, interfaces H/M et leurs conditionnements	Amplificateurs inverseur et non-inverseur, comparateurs, touche capacitive (Capsense)
Traiter	Système à microprocesseur, conversion A/N, conversion N/A, logiques combinatoire et séquentielle	Microcontrôleur 32 bits ARM, convertisseur A/D à approximation successive, convertisseur A/D Sigma-delta, convertisseurs D/A, logique combinatoire et séquentielle, compteur et Timer
Communiquer	Modules de communication entre système de traitement	Interfaces USB, RS232, LIN, I2C ..
Restituer	Affichage, génération de messages sonores	Afficheur LCD alphanumérique, graphique, afficheurs 7 segments ...

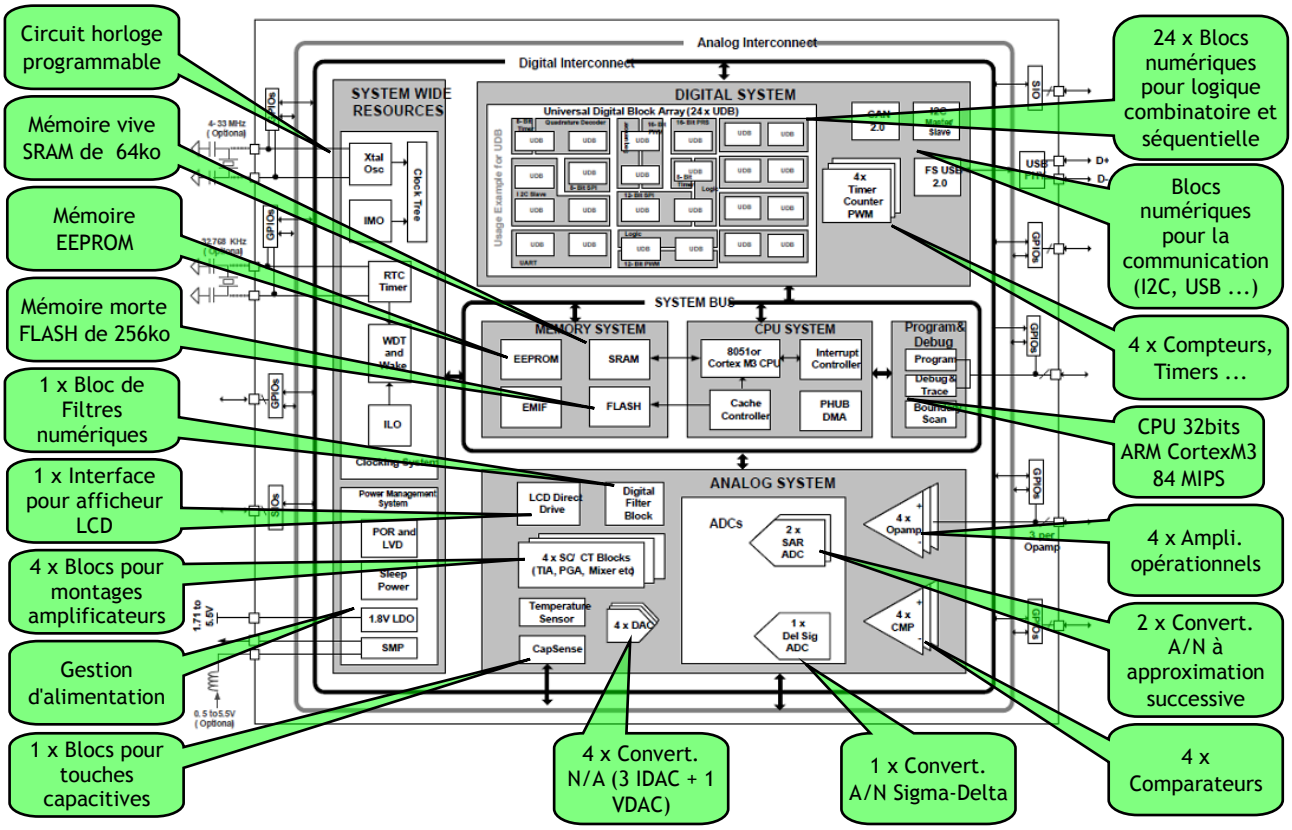
2.3 Chaîne d'énergie

Fonction	*Constituants*
Alimenter	Alimentation à découpage, régulateurs, convertisseurs électriques
Distribuer	Pont en H, transistors, relais ...
Convertir	Moteurs électriques
Transmettre	Réducteur, système poulie courroie ...

3 Le microcontrôleur « PSOC5 »

3.1 Description détaillée « CY8C5568AXI-060 »

Comme cela a été décrit précédemment, le PSOC 5 offre beaucoup de possibilités. Celui qui est utilisé par la suite a pour référence « CY8C5588AXI » et comporte 100 broches. Son boîtier est du type CMS (Composant Monté en Surface) et n'occupe qu'une surface de 2 cm² sur la carte électronique. Le rapport taille / puissance est très important pour le développement de systèmes embarqués autonomes. Voici le schéma illustrant ses multiples fonctionnalités :



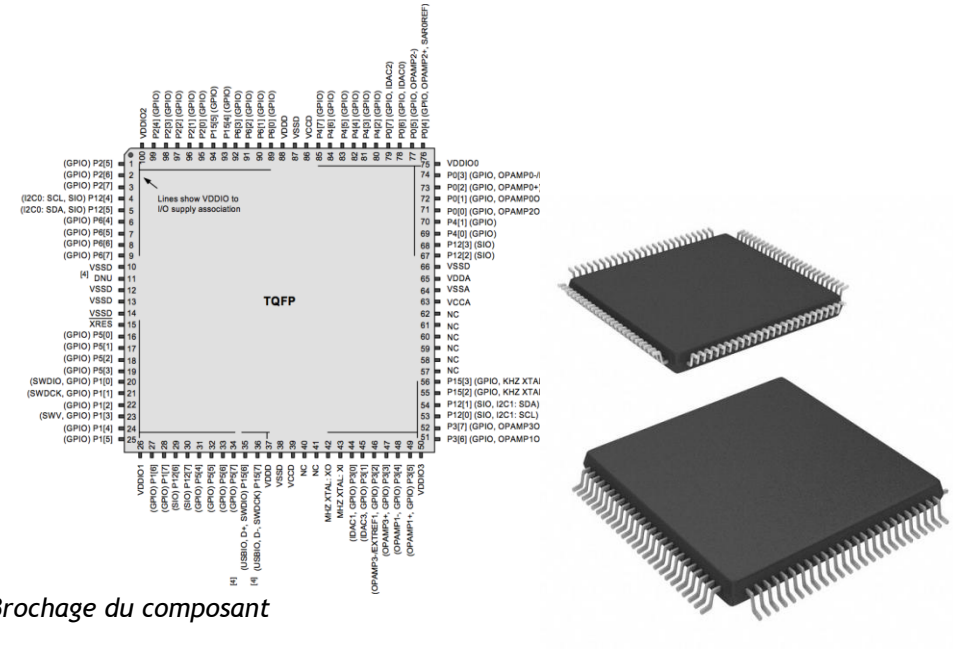
- Le processeur (CPU) :
  Cadencé avec une horloge de 67MHz, il offre une rapidité de calcul de 84Mips (84 Millions d'instructions par seconde). C'est un processeur ARM (Advanced Risc Machine) Cortex M3, 32 bits. Les données peuvent donc être traitées directement par « paquets » de 32 bits.

- La mémoire « morte » (256ko FLASH) :
  C'est une mémoire FLASH (comme les clefs USB ou les lecteurs MP3) de capacité 256ko (kilo octet). Cette mémoire contient le programme que le microprocesseur va exécuter.

- La mémoire « vive » (64ko SRAM) :
  C'est une mémoire SRAM (Static Random Access Memory – mémoire à accès aléatoire) qui permet de stocker les données utiles au programme (par exemple les variables déclarées dans le programme). Sa capacité est de 64Ko (64 kilo octet).

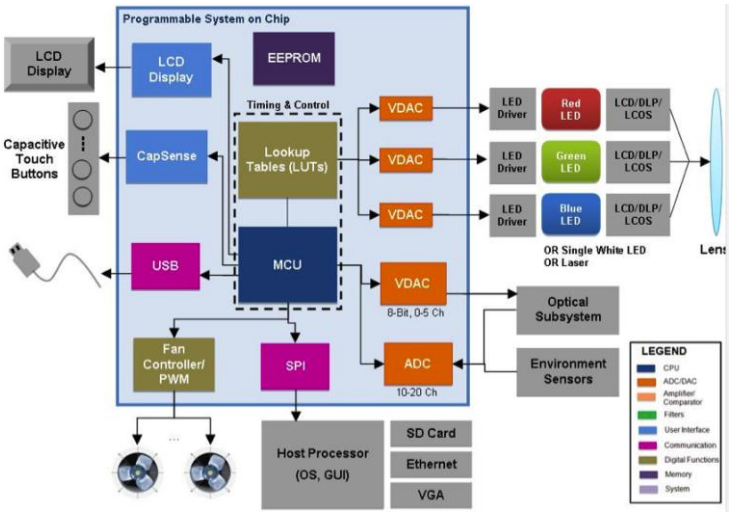
- Les entrées /sorties « I/O » :
  Au nombre de 72, elles offrent au microcontrôleur de nombreuses voies de communication avec l'extérieur. 62 entrées / sorties d'usage général (GPIO - General Purpose Input Output), 8 entrées/sorties « séries » (SIO – Serial Input Output) et 2 entrées/sorties pour la communication USB (USBIO – Universal Serial Bus Input Output).

3.1.1 Brochage du composant en boîtier TQFP :



Brochage du composant

3.1.2 Exemple d'application : vidéoprojecteur à LEDs



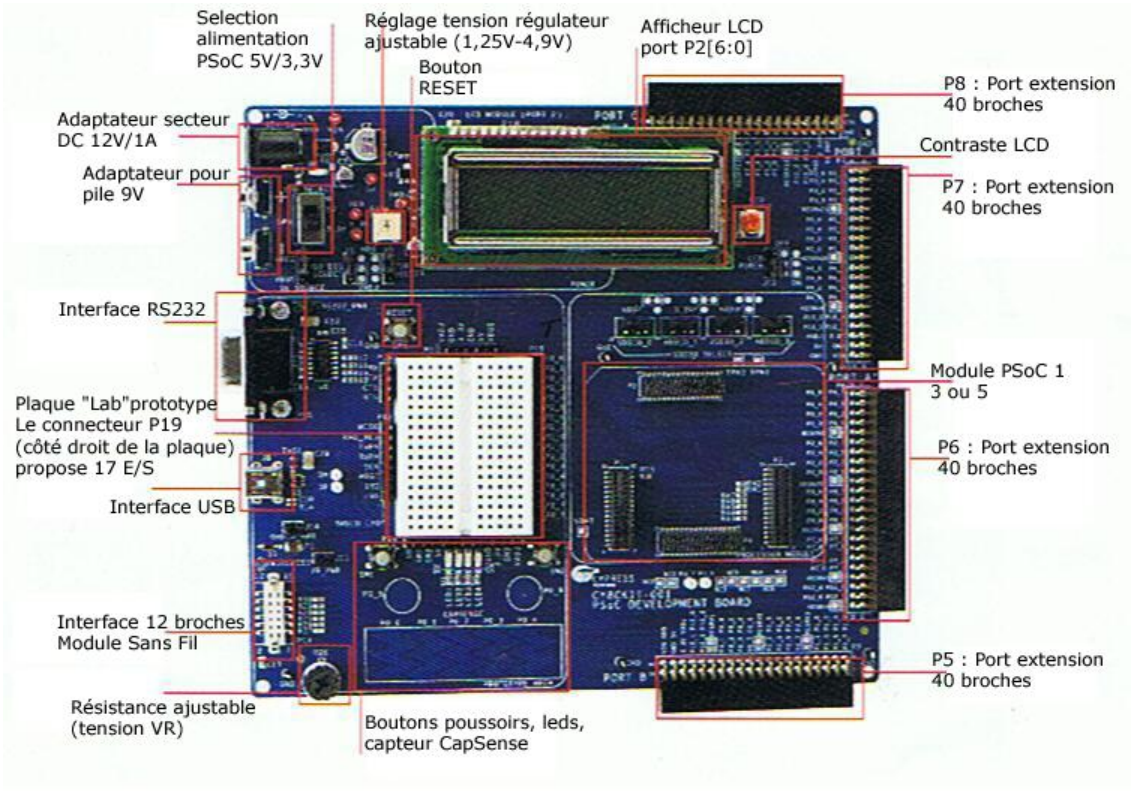
Dans cette application le microcontrôleur PSoC permet de :

- contrôler l'intensité dans les LEDs,
- contrôler la vitesse des ventilateurs,
- mesurer la température,
- assurer le dialogue entre l'appareil et l'utilisateur (Touches capacitives et afficheur LCD),
- communiquer avec le processeur hôte,
- assurer la liaison USB.

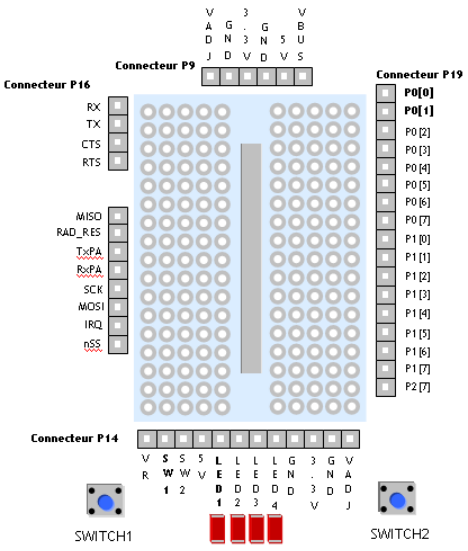
3.2 Le kit « CY8CKIT-001 »

Le kit sur lequel nous mettrons en œuvre le PSoC 5 est celui qui a pour référence « CY8CKIT-001 » et qui est constitué de 2 parties : d'une carte mère dotée de plusieurs éléments permettant de tester les fonctionnalités du PSoC et d'une carte fille (trois modèles différents selon la famille souhaitée de PSoC), équipée du microcontrôleur.

Un programmateur USB permet le transfert du programme compilé dans le composant PSoC situé sur la carte fille.



Carte mère du kit



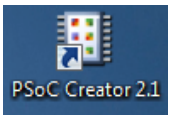
Zone de prototypage "Labdec"

Carte fille PSoC et programmateur USB

3.3 Logiciel « PSoC Creator »

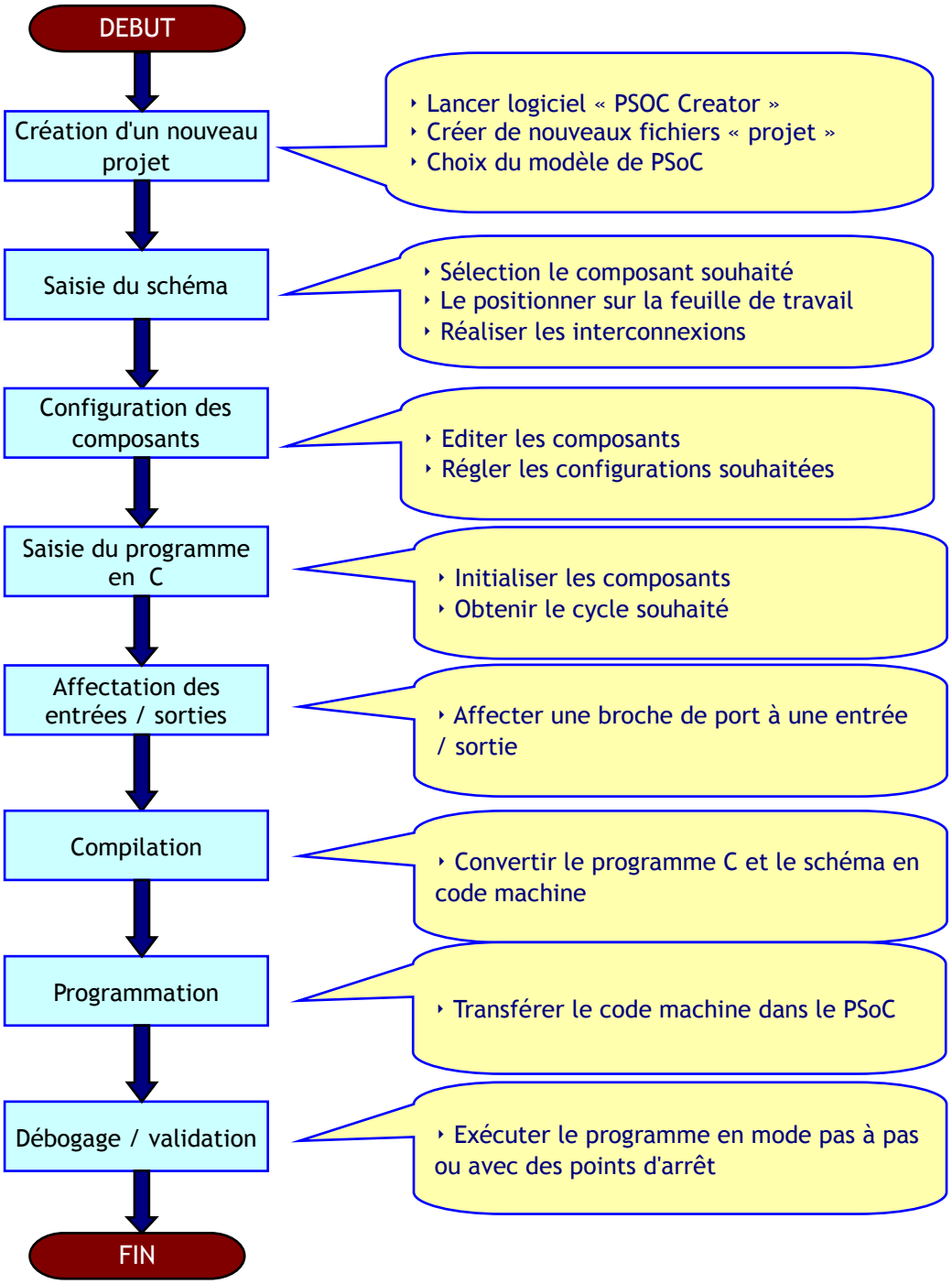
« Psoc Creator » (version 2.1) est un environnement de développement intégré (IDE) comprenant :

- un navigateur de projet,
- un éditeur de schéma,
- un éditeur de texte pour la saisie du programme,
- un compilateur C (GCC pour les PSoC 5),
- un gestionnaire d'affectation de broches de ports,
- Un module de programmation et de débogage.

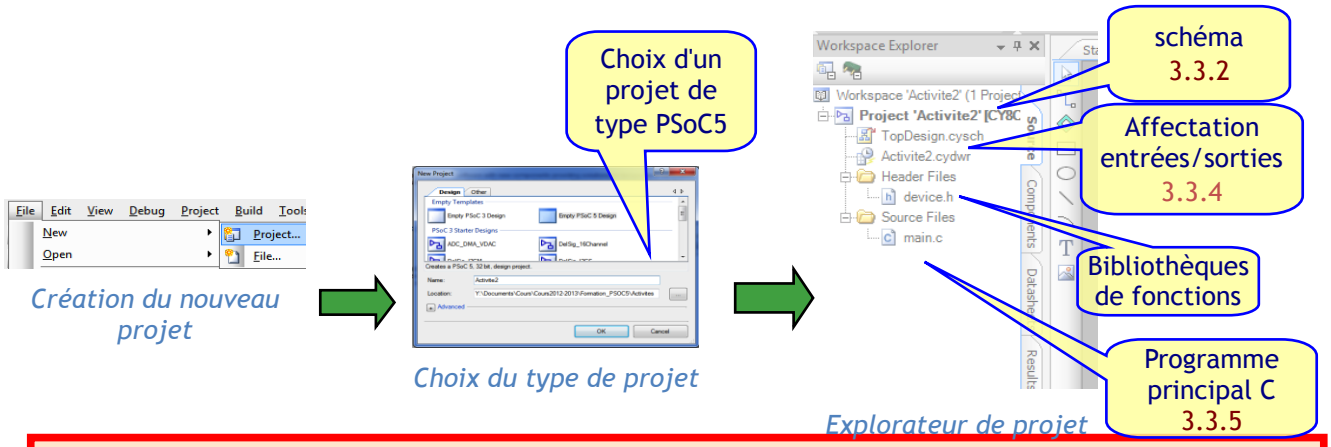


Il permet de développer et de déboguer toutes les applications à base de PSoC 3 et 5.

Voici la démarche de développement d'un projet à base de PSoC 5 :

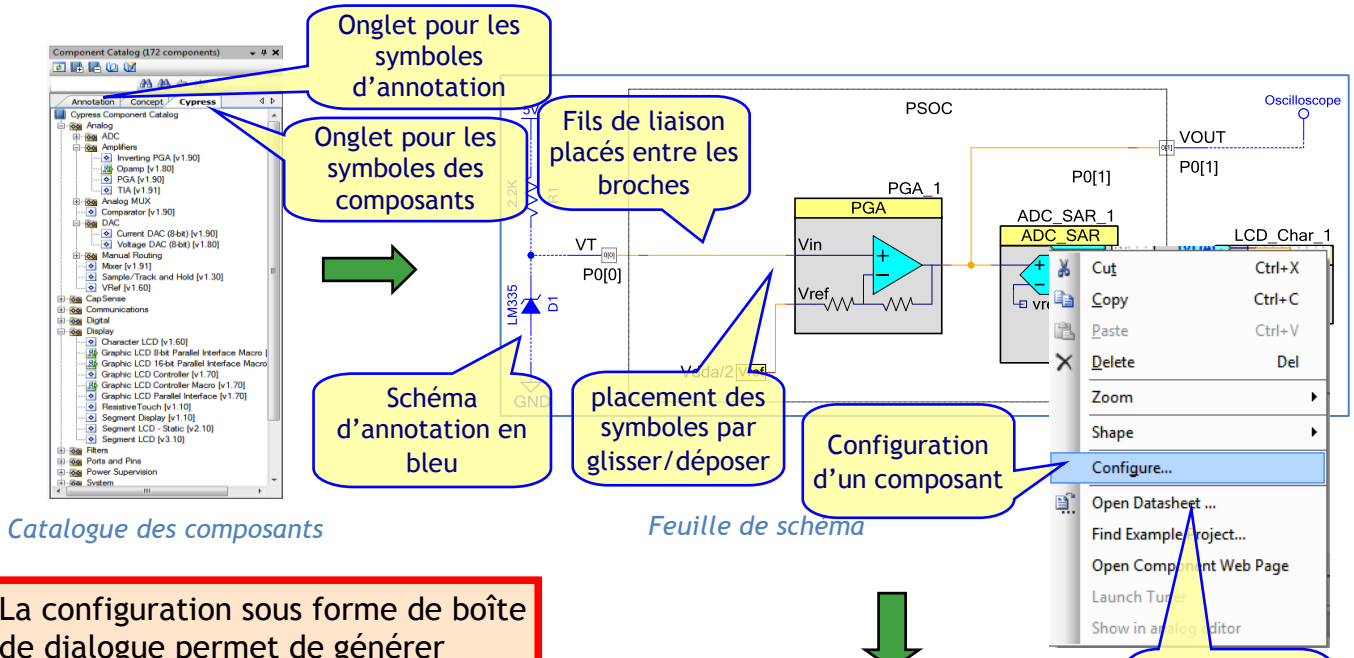


3.3.1 Création d'un nouveau projet

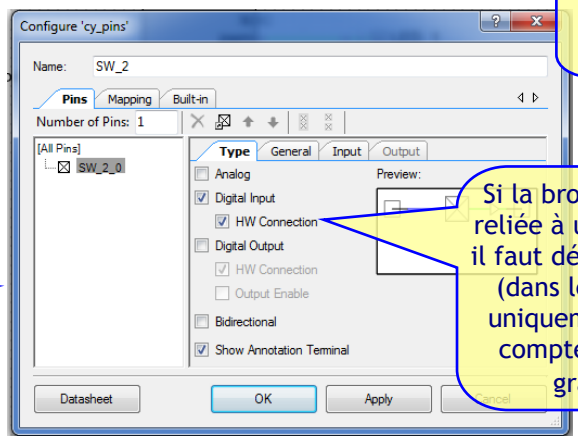


Dans l'explorateur de projet, un double-clic sur un fichier du projet permet de l'éditer. Les fichiers principalement éditables sont le schéma, le programme en C et l'affectation des entrées / sorties.

3.3.2 Saisie du schéma et configuration des composants



La configuration sous forme de boîte de dialogue permet de générer automatiquement le code d'initialisation d'un composant. Par la suite, dans l'édition du programme, il faut appeler la fonction associée dans la partie configuration.

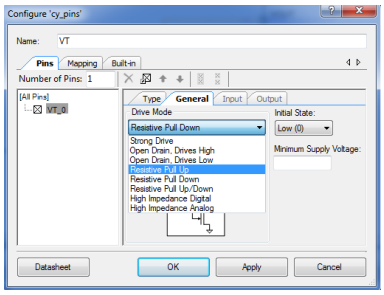


Documentation du composant en « pdf »

La configuration du composant se fait avec une boîte de dialogue disposant de cases à cocher, de listes déroulantes et de zones de saisies de valeurs

Si la broche n'est pas reliée à un composant, il faut décocher la case (dans le cas où elle uniquement prise en compte par le programme).

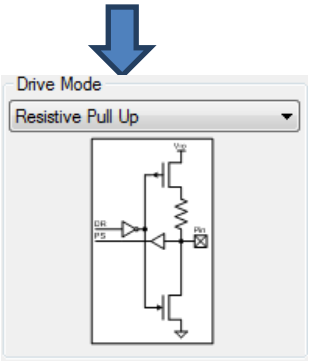
3.3.3 Configuration des entrées et sorties



Les **entrées** et **sorties** du PSOC peuvent être de deux natures : **analogique** ou **logique**. Le symbole utilisé permet de définir ces paramètres mais dans le cas du mode logique plusieurs configurations sont possibles afin de répondre à différentes contraintes.

Si l'élément connecté sur une broche d'**entrée** est du type contact ouvert ou collecteur/drain ouvert, il est nécessaire d'ajouter une résistance de tirage à la masse ou au plus de l'alimentation afin de fixer le niveau logique en l'absence d'information.

La configuration d'une broche de port s'obtient en lançant la commande « **configure** » puis en sélectionnant le « **Drive mode** » dans l'onglet « **General** ». Une fois validé, un petit schéma illustre le mode configuré.

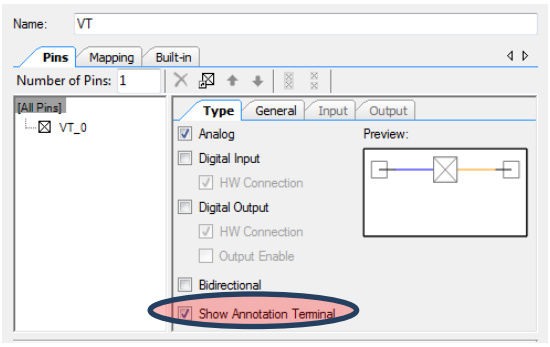


Les configurations possibles des entrées sont celles-ci :

Type de configuration de l'entrée	***Choix « Drive mode »***
Niveau bas au repos	Resistive pull-down
Niveau haut au repos	Resistive pull-up
Niveau intermédiaire (VDD / 2)	Resistive pull-up / down
Non défini	High impedance digital / analog

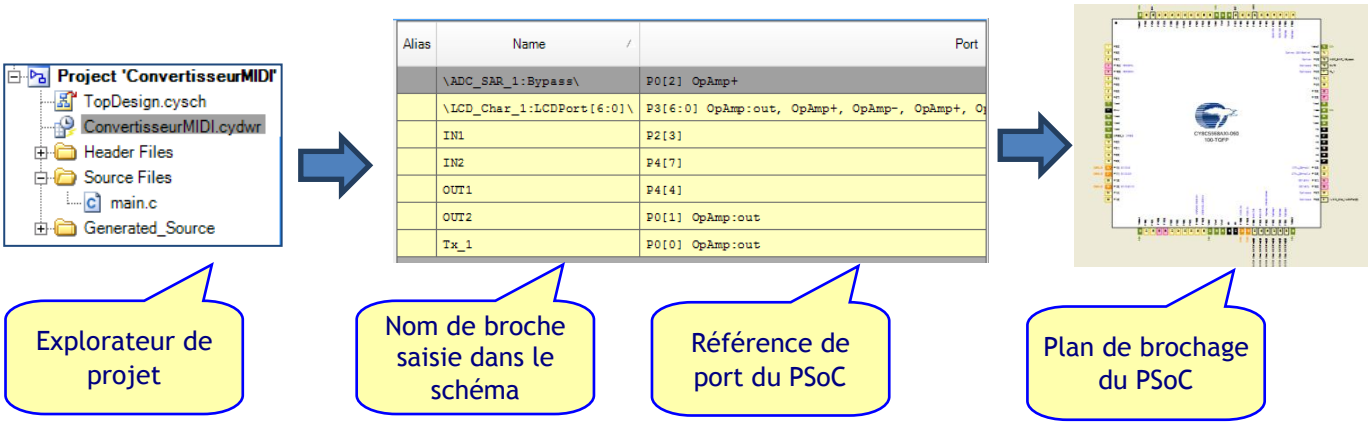
Les configurations possibles des sorties sont celles-ci :

Type de configuration de la sortie	***Choix « Drive mode »***
Imposer le niveau bas	Open drain, drive low
Imposer le niveau haut	Open drain, drive high
Imposer les niveaux haut et bas	Strong



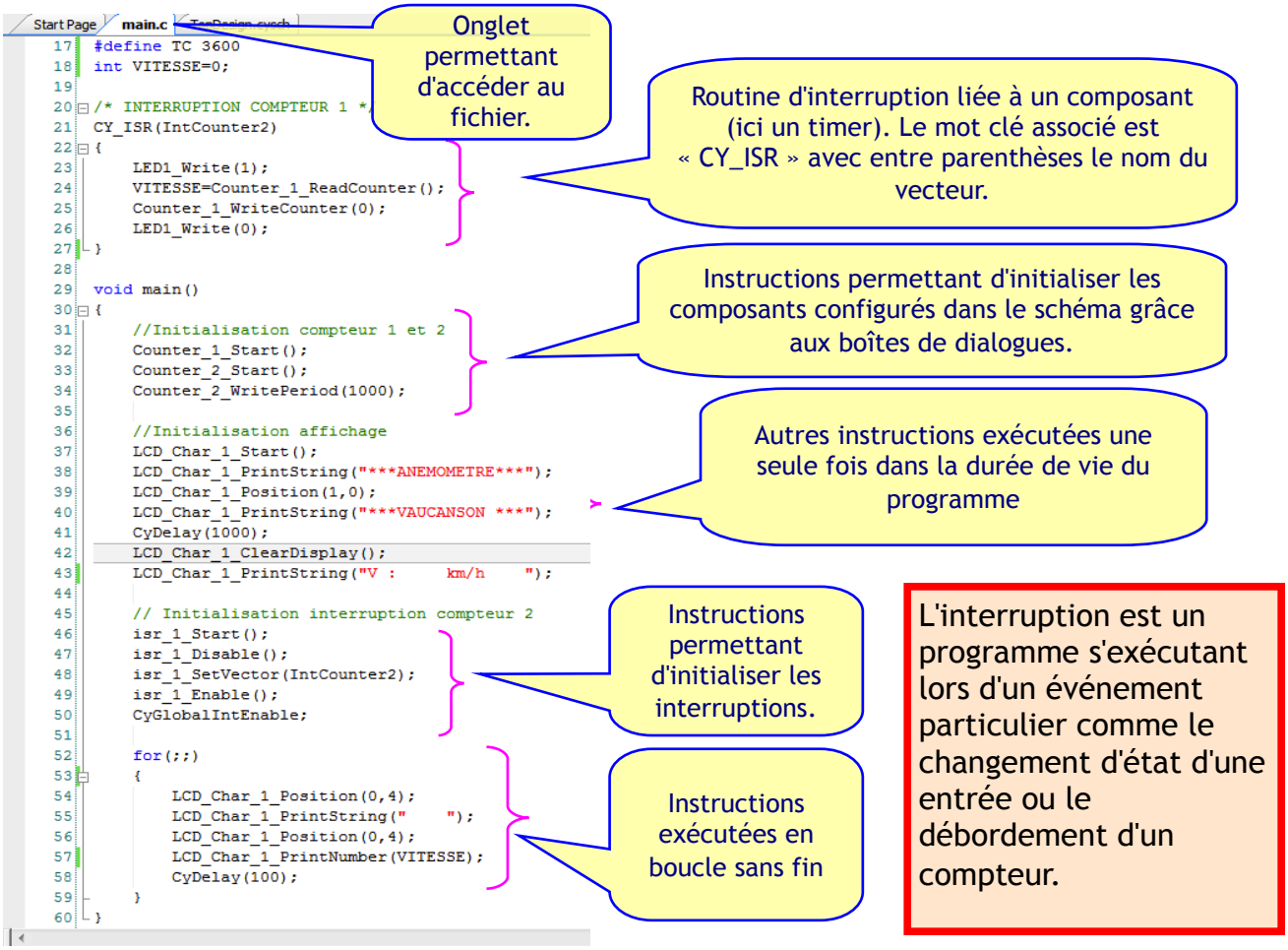
Sur le schéma il est possible de rajouter des symboles d'annotation afin de d'indiquer les connexions externes des broches du PSoC. Pour que cela se fasse correctement il est possible de configurer les broches en mode « Show annotation terminal ».

3.3.4 Affectation des entrées / sorties



Dans l'explorateur de projet, on sélectionne le fichier ayant l'extension « .cydwr » afin de définir les affectations des broches d'entrées / sorties. Dans le tableau qui s'affiche, on choisit un port spécifique et distinct pour chaque nom de broche saisie précédemment sur le schéma. On peut observer cette affectation sur le plan de brochage du PSoC.

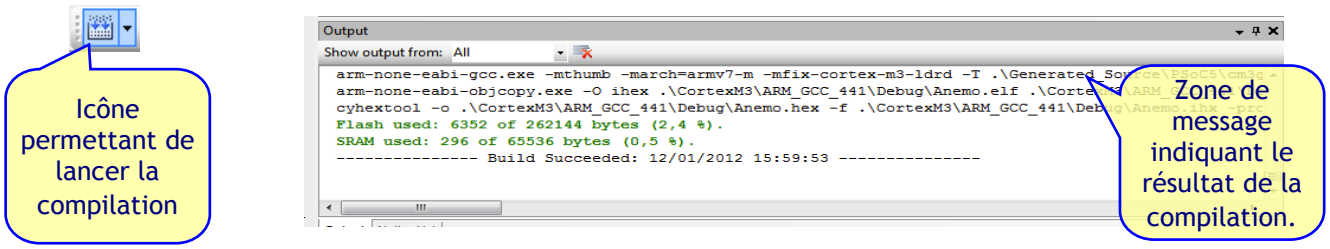
3.3.5 Saisie du programme en « C »




```
Start Page | main.c
17  #define TC 3600
18  int VITESSE=0;
19
20  /* INTERRUPTION COMPTEUR 1 */
21  CY_ISR(IntCounter2)
22  {
23      LED1_Write(1);
24      VITESSE=Counter_1_ReadCounter();
25      Counter_1_WriteCounter(0);
26      LED1_Write(0);
27  }
28
29  void main()
30  {
31      //Initialisation compteur 1 et 2
32      Counter_1_Start();
33      Counter_2_Start();
34      Counter_2_WritePeriod(1000);
35
36      //Initialisation affichage
37      LCD_Char_1_Start();
38      LCD_Char_1_PrintString("***ANEMOMETRE***");
39      LCD_Char_1_Position(1,0);
40      LCD_Char_1_PrintString("***VAUCANSON ***");
41      CyDelay(1000);
42      LCD_Char_1_ClearDisplay();
43      LCD_Char_1_PrintString("V :      km/h     ");
44
45      // Initialisation interruption compteur 2
46      isr_1_Start();
47      isr_1_Disable();
48      isr_1_SetVector(IntCounter2);
49      isr_1_Enable();
50      CyGlobalIntEnable;
51
52      for(;;)
53      {
54          LCD_Char_1_Position(0,4);
55          LCD_Char_1_PrintString("    ");
56          LCD_Char_1_Position(0,4);
57          LCD_Char_1_PrintNumber(VITESSE);
58          CyDelay(100);
59      }
60  }
```

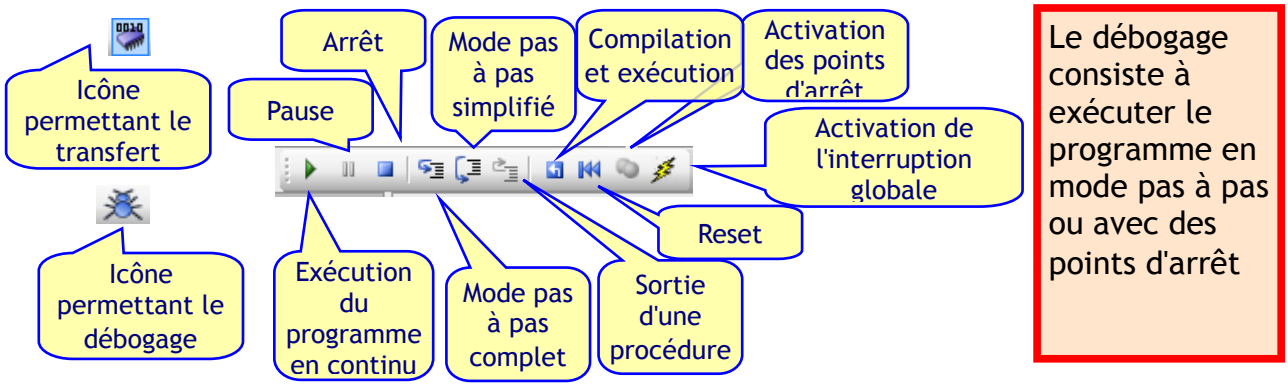
L'interruption est un programme s'exécutant lors d'un événement particulier comme le changement d'état d'une entrée ou le débordement d'un compteur.

3.3.6 Compilation



La compilation est une opération qui peut durer jusqu'à 1 ou 2 minutes. Cette longueur est due à la configuration des composants qui correspond à un routage de différents blocs élémentaires présents dans le PSoC.

3.3.7 Transfert du programme et débogage

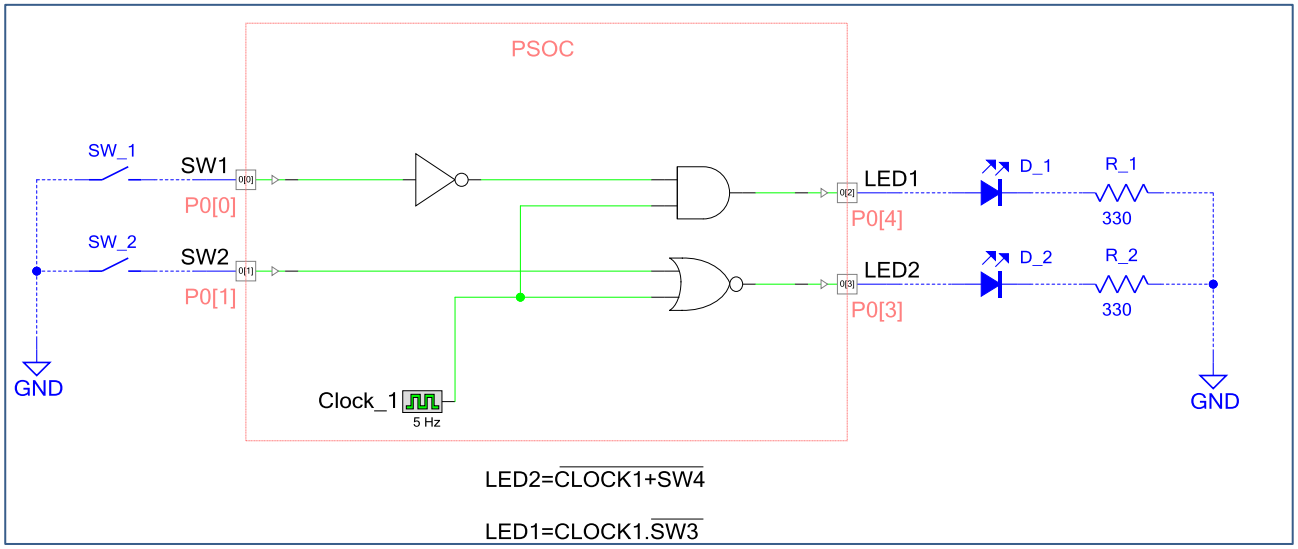


Le débogage consiste à exécuter le programme en mode pas à pas ou avec des points d'arrêt

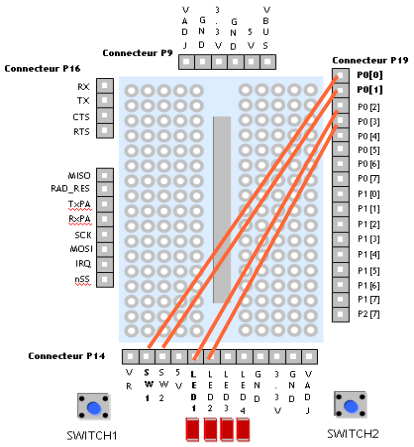
4 Initiation à la programmation des « PSOC 5 » avec « PSOC Creator »

4.1 Premier projet « clignoteur à LED »

Ce premier projet permet de faire clignoter 2 LEDs à partir d'une commande faite par des boutons poussoirs. Il n'y a pas de programme littéral en C. La structure fait appel à un générateur de signaux d'horloge et à des portes logiques.

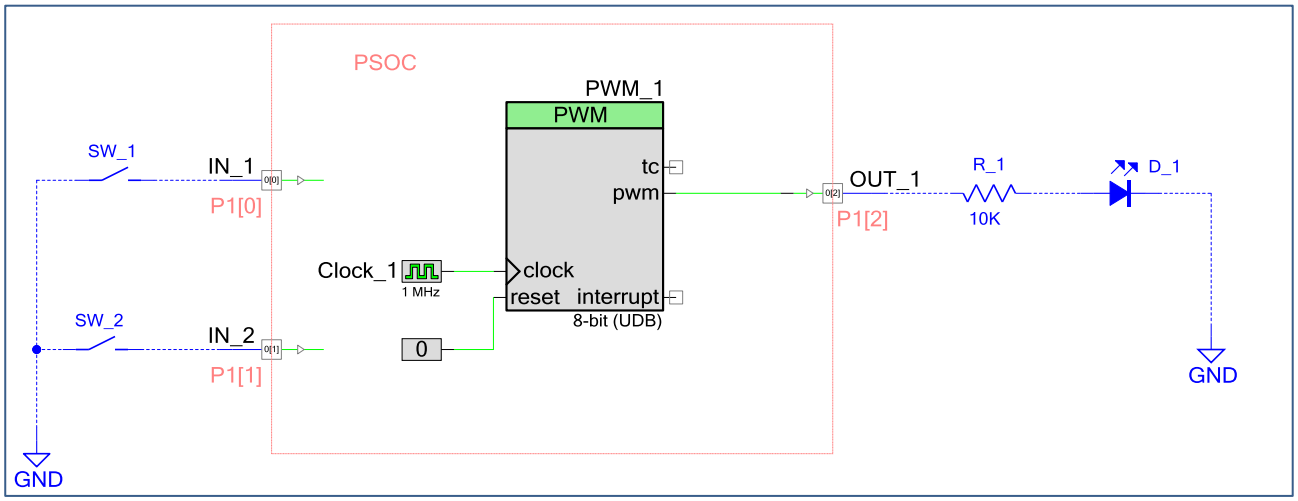


1) Créer un nouveau projet « Activite01 » avec un PSoC 5 ayant pour référence « CY8C5568AXI-60 ».
2) Saisir le schéma précédent.
3) Configurer les entrées en mode « Pull-up ».
4) Définir les affectations des broches d'entrées / sorties.
5) Câbler (hors tension) le kit selon le schéma puis vérifier le fonctionnement.
6) Compiler le projet puis transférer le programme dans le PSoC.



4.2 Variation de la luminosité d'une LED par « PWM »

Ce projet a pour but de faire varier la luminosité d'une LED par modulation de largeur d'impulsion en agissant sur 2 boutons poussoirs. La structure fait appel à un TIMER PWM en mode 8 bits (256 valeurs de comptage). Cette fois ci il y a un programme qui gère la valeur de consigne en fonction des appuis sur les boutons poussoir.

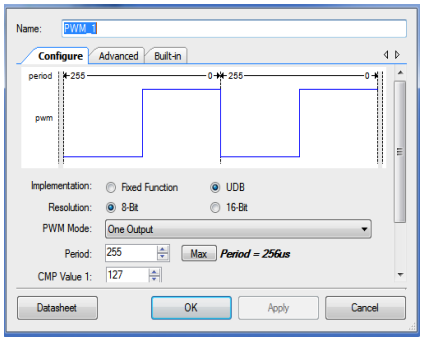


```c
#include <device.h>

void main()
{
    uint8 Value=0;
    /* Place your initialization/startup code here (e.g. MyInst_Start()) */
    PWM_1_Start();
    /* CyGlobalIntEnable; */ /* Uncomment this line to enable global interrupts. */
    for(;;)
    {
        if((IN_1_Read()==0) && (Value<255)) Value++;
        else if ((IN_2_Read()==0) && (Value>0)) Value--;
        PWM_1_WriteCompare(Value);
        CyDelay(10);
    }
}
```

Programme en langage C

1) Créer un nouveau projet « activite02 » avec un PSoC 5 ayant pour référence « CY8C5568AXI-60 ».
2) Saisir le schéma puis le programme précédent.
3) Configurer le Timer « PWM_1 et les entrées en mode « Pull-up ».
4) Définir les affectations des broches d'entrées / sorties.
5) Câbler (hors tension) le kit selon le schéma puis vérifier le fonctionnement.
6) Compiler le projet puis transférer le programme dans le PSoC.



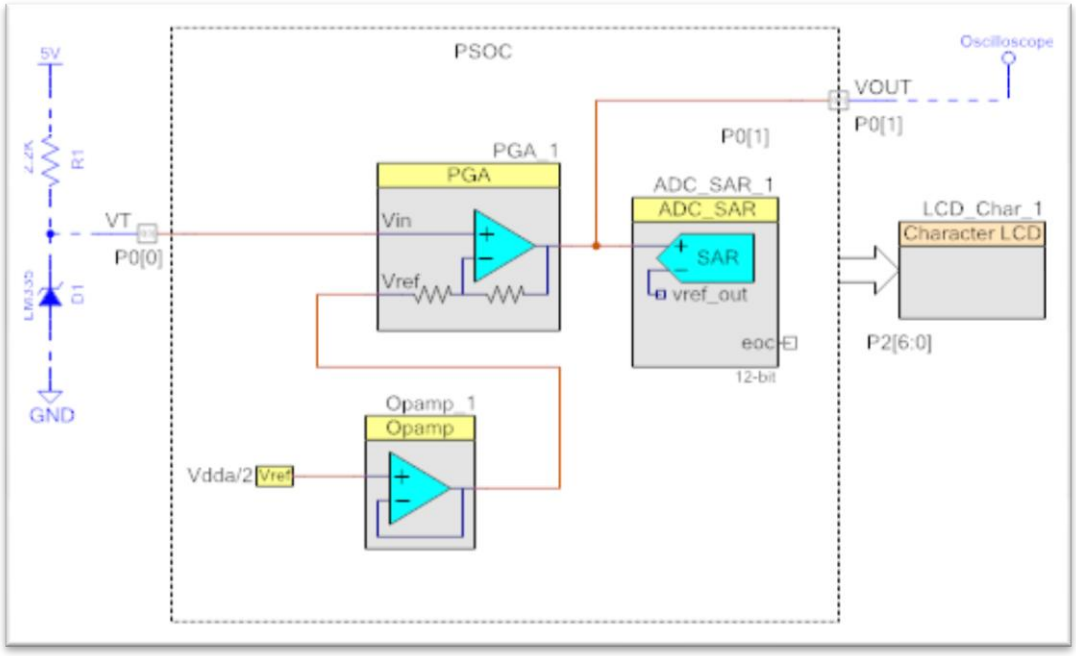
Configuration du compteur PWM

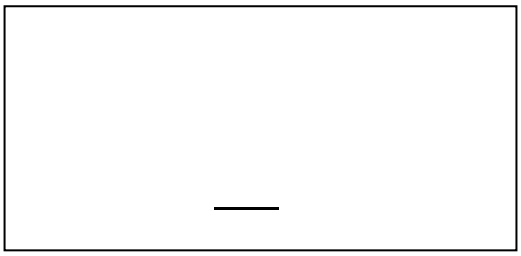
4.3 Thermomètre avec capteur analogique « LM335 »

Ce projet a pour but d'afficher la température ambiante en utilisant un capteur analogique « LM335 ». Ce capteur est constitué d'un semi-conducteur dont la caractéristique est de fournir une tension directement proportionnelle à la température en Kelvin. **Le coefficient de proportionnalité est de 10mV par Kelvin.**

Pour effectuer la mesure de la température en degrés Celsius le microcontrôleur doit effectuer les opérations suivantes :

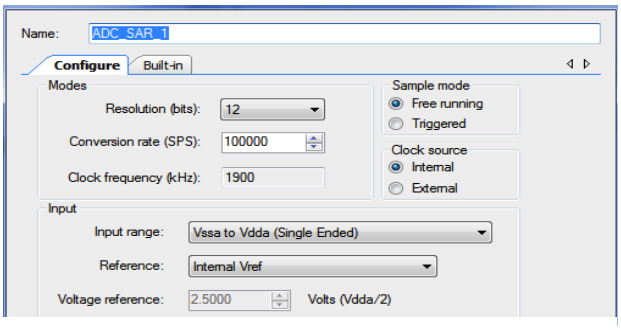
- Amplification de la tension issue du capteur afin d'obtenir une plage dynamique plus importante (le double),
- conversion de la tension amplifiée en grandeur numérique « N » codée sur 12 bits,
- mise à l'échelle de grandeur numérique afin d'obtenir une température en degrés Celsius « TC »
- Affichage du résultat.





Relations

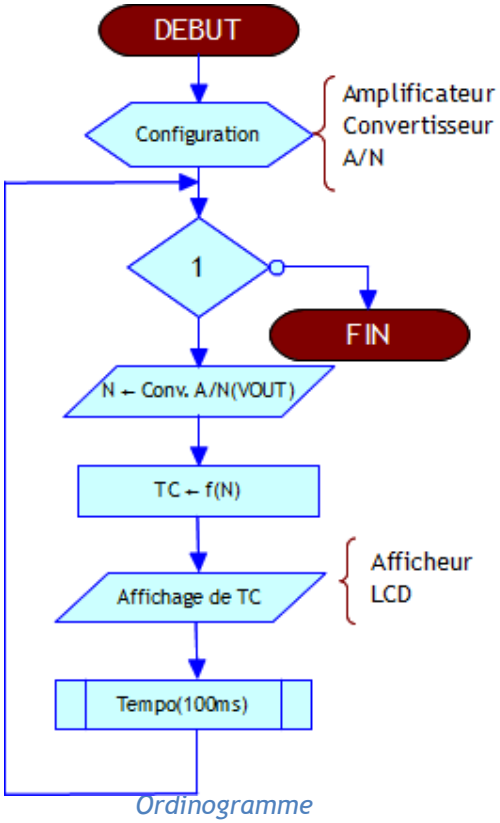
Brochage du capteur



Configuration du convertisseur A/N

1) Créer un nouveau projet « activite03 » avec un PSoC 5 ayant pour référence « CY8C5568AXI-060 ».
2) Saisir le schéma précédent.
3) Configurer les composants (amplificateur et convertisseur A/N) afin de répondre aux exigences (amplification de 2).
4) Définir les affectations des broches d'entrées / sorties.
5) Déterminer l'expression de « N » en fonction de la température « TC », puis l'expression inverse « TC = f (N) ».
6) A partir de l'ordinogramme, saisir le programme en langage C.

   *<u>Consignes</u> : On utilisera deux variables de type « float » **N** et **TC**. Une instruction correspondant à l'expression « TC = f(N) » permettra de retrouver la température avant de l'afficher. La phase de configuration permettra d'initialiser les composants paramétrés auparavant.*
7) Câbler (hors tension) le kit selon le schéma puis vérifier le fonctionnement.
8) Compiler le projet puis transférer le programme dans le PSoC.



Ordinogramme

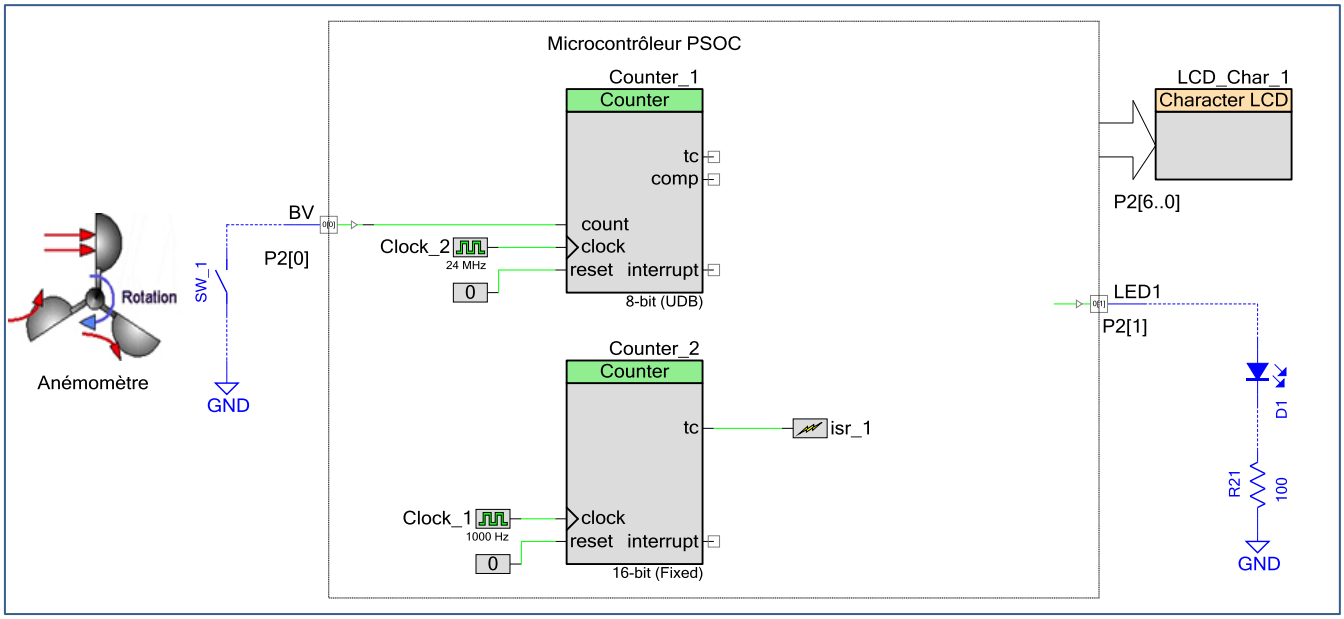
4.4 Anémomètre numérique avec capteur « ILS »

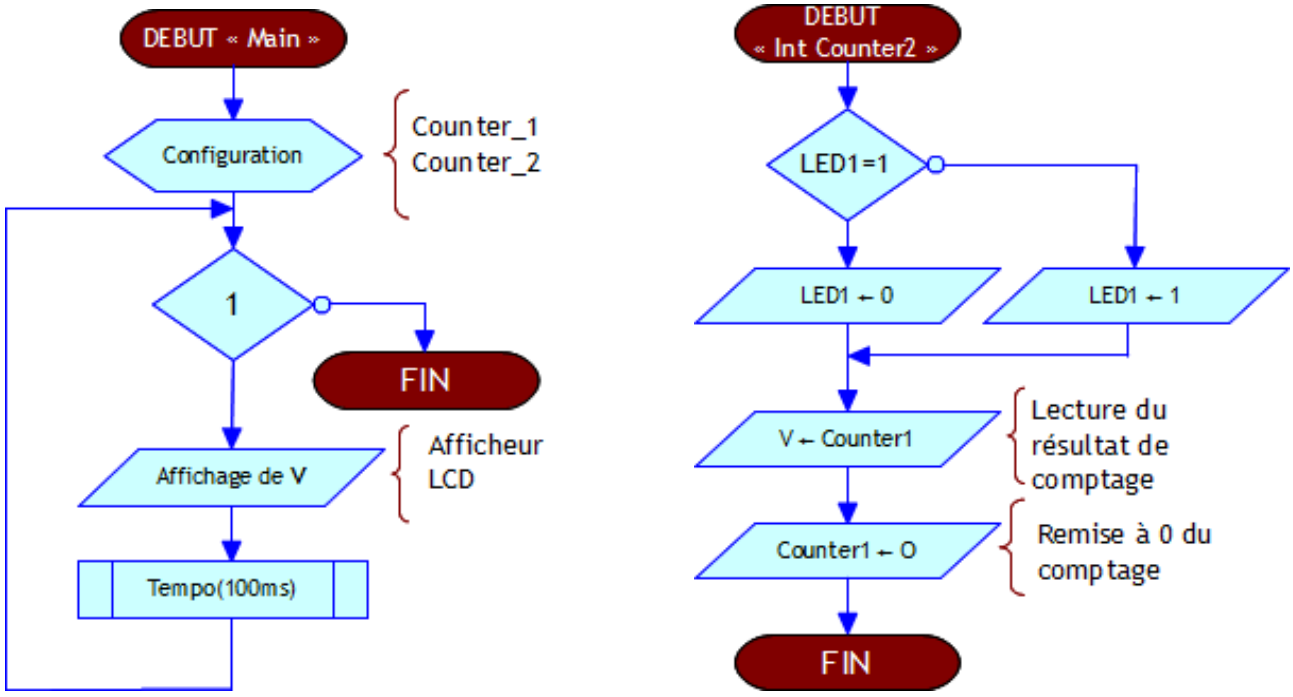
Ce projet a pour but d'afficher la vitesse du vent acquise grâce à un anémomètre à coupelles. Ce dernier fournit un signal électrique dont la fréquence est proportionnelle à la vitesse du vent. Un détecteur magnétique, placé sur la partie fixe située autour de l'axe de rotation, détecte le passage d'un aimant. Ce dernier est fixé sur la partie tournante entrainée par les 3 coupelles. La vitesse du vent est définie comme ceci : ***« le nombre d'impulsions fournies par le capteur en 3,6s correspond à la vitesse en km/h***.

Pour effectuer la mesure de la vitesse il suffit d'effectuer en boucle et dans l'ordre les opérations suivantes :

- compter le nombre d'impulsions délivrées par le capteur,
- au bout d'un temps fixé « T1 », mémoriser le résultat de comptage,
- remettre à zéro le comptage.

Les deux dernières opérations sont exécutées par un programme d'interruption (signal **isr_1**) déclenché par le compteur « Counter_2 » à intervalles réguliers (toutes les 3,6 s). Le comptage des impulsions est effectué par le compteur « Counter_1 »
Une LED « D1 » change d'état à chaque exécution de l'interruption.





Ordinogrammes du programme principal et du programme d'interruption

1) Créer un nouveau projet « activite04 » avec un PSoC 5 ayant pour référence « CY8C5568AXI-060 ».
2) Saisir le schéma précédent.
3) Définir les affectations des broches d'entrées / sorties.
4) Configurer les composants (les deux compteurs) afin de répondre aux contraintes décrites précédemment.
5) A partir de l'ordinogramme, saisir les deux programmes en langage C.

Consignes :

- La configuration permettra d'initialiser les deux compteurs ainsi que l'interruption :

```
// Initialisation interruption compteur 2
    isr_1_Start();
    isr_1_Disable();
    isr_1_SetVector(IntCounter2);
    isr_1_Enable();
    CyGlobalIntEnable;
```

- On utilisera une variable globale entière« V » indiquant la vitesse du vent.
- Le programme d'interruptions est défini avant le programme principal selon le code suivant :

```
/* INTERRUPTION COMPTEUR 1 */
CY_ISR(IntCounter2)
{
      …
      …
      …

}
```

7) Câbler (hors tension) le kit selon le schéma puis vérifier le fonctionnement (utilisation d'un GBF pour simuler l'anémomètre).
8) Compiler le projet puis transférer le programme dans le PsoC.