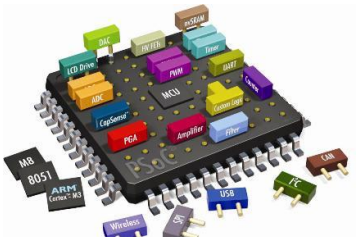
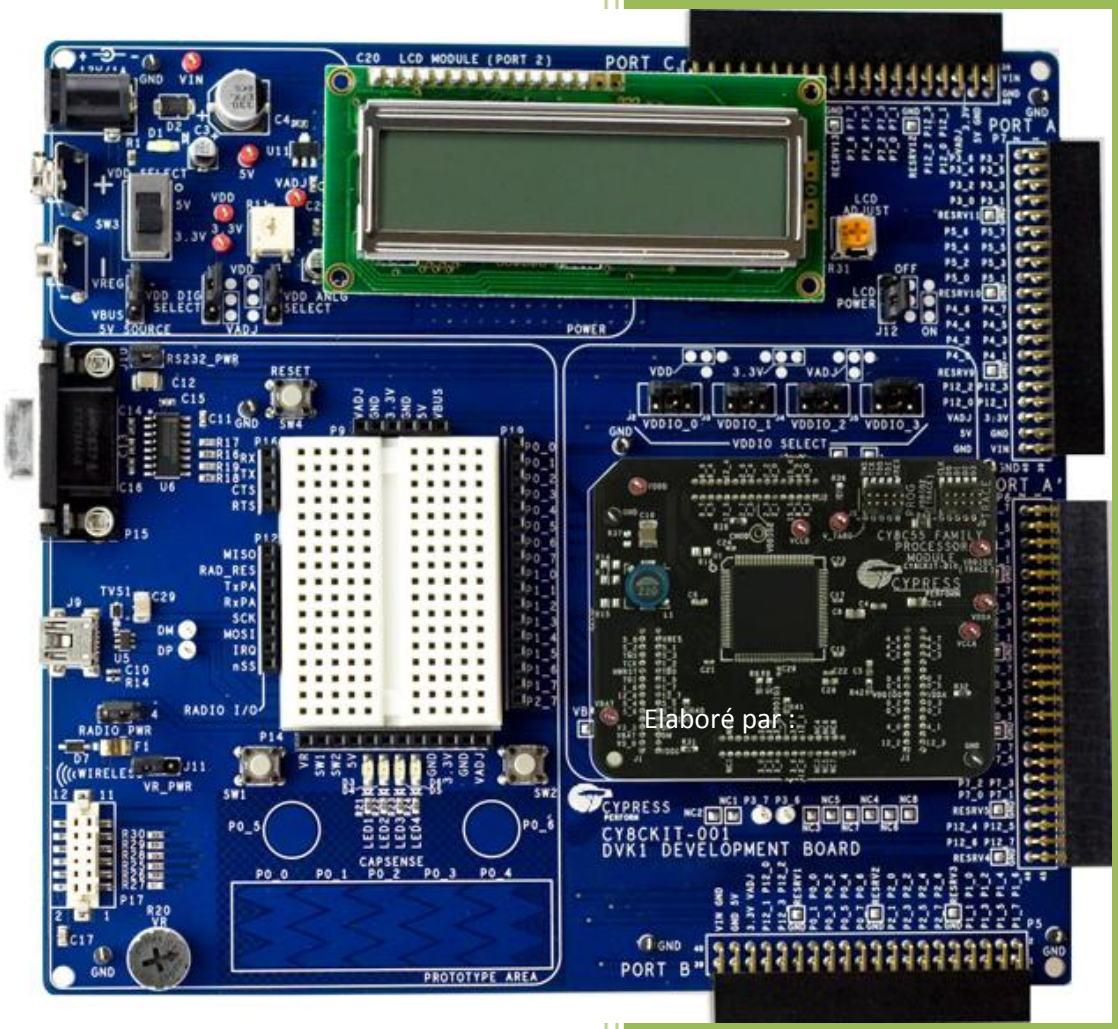
Ministères de l'enseignement supérieur
Institut Supérieur des Etudes Technologiques de Gabès



Année Universitaire 2014-2015

SUPPORT DE COURS SYSTEMES EMBARQUES



Elaboré par :

TAYARI LASSAAD

MAITRE TECHNOLOGUE A ISET GABES

E-mail :lassaad.tayari@isetn.rnu.tn

Chapitre I

Introduction aux systèmes embarqués

CONTENU

Chapitre I

Introduction aux systèmes embarqués

I.Introduction : Qu'est-ce qu'un système embarqué? Qu'est-ce que l'embarqué ?

Il suffit de regarder autour de soi au quotidien pour voir et avoir la réponse sous ses yeux.

Vous êtes réveillé le matin par votre radioréveil ; c'est un système embarqué.

Vous programmez votre machine à café pour avoir un bon petit serré; c'est un système embarqué.

Vous allumez la télévision et utilisez votre télécommande ; ce sont des systèmes embarqués.

Vous prenez votre voiture et la voix du calculateur vous dit que vous n'avez pas mis votre ceinture; c'est un système embarqué.

Vous appelez votre ami avec votre téléphone portable pour signaler que vous serez en retard; c'est un système embarqué.

On peut continuer à énumérer tous les systèmes embarqués croisés sans le savoir au cours d'une journée.

Bref, les systèmes embarqués nous entourent et nous sommes littéralement envahis par eux, fidèles au poste et prêts à nous rendre service. On en croise des dizaines par jour sans le savoir.

Ils sont donc partout, discrets, efficaces et dédiés à ce à quoi ils sont destinés. Omniprésents, ils le sont déjà et le seront de plus en plus. On parle en fait d'informatique (et d'électronique).

Ils sont bourrés d'électronique plus ou moins complexe et d'informatique plus ou moins évoluée.

Essayons de donner une définition plus précise d'un système embarqué :

Un système embarqué peut être défini comme un système électronique et informatique autonome, qui est dédié à une tâche bien précise.

Il ne possède généralement pas des entrées/sorties standards et classiques comme un clavier ou un écran d'ordinateur. Le système matériel et l'application sont intimement liés, le logiciel embarqué étant enfoui, noyé dans le matériel. Le matériel et le logiciel ne sont pas aussi facilement discernables comme dans un environnement de travail classique de type ordinateur PC.

L'embarqué est un terme plus général qui regroupe plusieurs notions selon le contexte :

- Le marché des systèmes embarqués.
- Les systèmes embarqués par abus de langage.

II.Caractéristiques principales d'un système embarqué

Les principales caractéristiques d'un système embarqué sont les suivantes :

- C'est un système principalement numérique.
- Il met en œuvre généralement un processeur.

• Il exécute une application logicielle dédiée pour réaliser une fonctionnalité précise et n'exécute donc pas une application scientifique ou grand public traditionnelle.

• Il n'a pas réellement de clavier standard (Bouton Poussoir, clavier matriciel...). L'affichage est limité (écran LCD…) ou n'existe pas du tout.

• Ce n'est pas un PC en général mais des architectures similaires (x86) basse consommation sont de plus en plus utilisées pour certaines applications embarquées.

De ce constat, on peut voir :

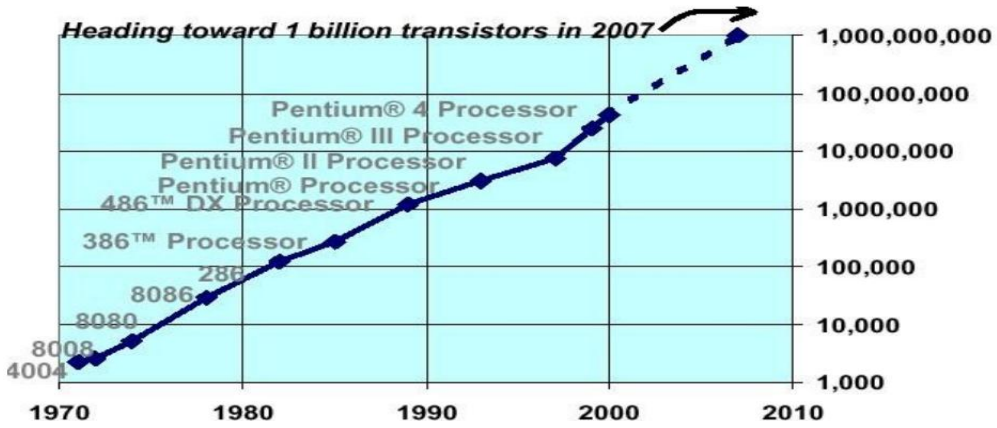
• Qu'un PC standard peut exécuter tout type d'applications car il est généraliste alors qu'un système embarqué n'exécute qu'une application dédiée.

• Que l'interface IHM peut être aussi simple qu'une led qui clignote ou aussi complexe qu'un cockpit d'avion de ligne.

• Que des circuits numériques ou des circuits analogiques sont utilisés en plus pour augmenter les performances du système embarqué ou sa fiabilité.

L'omniprésence des systèmes embarqués dans notre vie est liée à la révolution numérique opérée dans les années 1970 avec l'avènement des processeurs. Les processeurs, de plus en plus rapides, puissants et bon marché ont permis cette révolution et aussi le boom du marché de l'embarqué. Ceci se confirme au travers de la loi empirique de Gordon Moore, cofondateur d'Intel, qui stipule que pour une surface de silicium donnée, on double le nombre de transistors intégrés tous les 18 mois !

La figure 1.1 montre cette évolution inexorable.



En 1999, il a été vendu pour le marché de l'embarqué :

- 1,3 milliard de processeurs 4 bits.
- 1,4 milliard de processeurs 8 bits.
- 375 millions de processeurs 16 bits.
- 127 millions de processeurs 32 bits.
- 3,2 millions de processeurs 64 bits.

A côté de cela, à cette époque, il a été vendu seulement 108 millions de processeurs (famille x86) pour le marché du PC grand public !

Pour 2004, il a été vendu environ 260 millions de processeurs pour le marché du PC grand public à comparer aux 14 milliards de processeurs tout type confondu (microprocesseur, microcontrôleur, DSP) pour le marché de l'embarqué.

Les chiffres parlent d'eux-mêmes. Le marché du processeur pour les PC grand public n'est que la partie émergée de l'iceberg et n'est rien par rapport au marché de l'embarqué.

On peut aussi tirer le constat actuel suivant :

- Moins de 2 % des processeurs vendus sont pour le marché du PC contre 98 % pour l'embarqué.

On utilise massivement pour le PC grand public un système d'exploitation commercial bien connu.

- Pour les 98 % autres processeurs vendus, on utilisera généralement un autre système d'exploitation. On trouvera ici dans 60 % des cas un système d'exploitation commercial spécialisé pour l'embarqué (VxWorks, QNX...). Pour le reste, il s'agit d'un système d'exploitation maison (home made) mais de plus en plus optent pour des systèmes d'exploitation libres comme Linux pour l'embarqué.
- Moins de 10 % des processeurs vendus sont des processeurs 32 bits pour près de 31 % du chiffre d'affaire sur les processeurs. Cette part du chiffre d'affaire est de 48 % pour 2008 : cela montre la migration rapide vers ces processeurs 32 bits dans l'embarqué, condition nécessaire pour pouvoir mettre en œuvre Linux.
- Si l'on regarde le prix moyen d'un processeur tout type confondu, on arrive à 6 USD par unité à comparer au prix moyen de 300 USD par unité pour un processeur pour PC. Le marché du processeur pour PC est très faible en volume mais extrêmement lucratif !

Ces quelques chiffres permettent bien de prendre conscience de l'importance du marché de l'embarqué.

Les grands secteurs de l'embarqué concernent les domaines suivants :

- Jeux et calcul général : application similaire à une application de bureau mais empaquetée dans un système embarqué : jeux vidéo, set top box...
- Contrôle de systèmes : automobile, process chimique, process nucléaire, système de navigation...
- Traitement du signal : radar, sonar, compression vidéo...
- Communication et réseaux : transmission d'information et commutation, téléphonie, Internet... La figure 1.2 présente les caractéristiques principales d'un système embarqué typique.

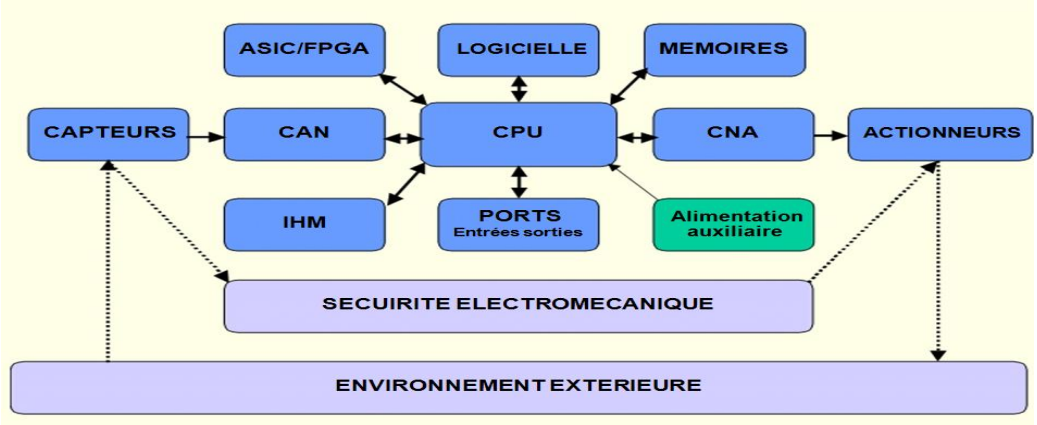


Figure 1.2 : Système embarqué typique

On retrouve en entrée des capteurs généralement analogiques couplés à des convertisseurs A/N.

On retrouve en sortie des actionneurs généralement analogiques couplés à des convertisseurs N/A. Au milieu, on trouve le calculateur mettant en œuvre un processeur embarqué et ses périphériques d'E/S. Il est à noter qu'il est complété généralement d'un circuit FPGA jouant le rôle de coprocesseur afin de proposer des accélérations matérielles au processeur.

Sur ce schéma théorique se greffe un paramètre important: le rôle de l'environnement extérieur.

Contrairement au PC ronronnant bien au chaud dans un bureau, un système embarqué doit faire face à des environnements plus hostiles. Il doit faire face à un ensemble de paramètres agressifs :

- Variations de la température.
- Vibrations, chocs.
- Variations des alimentations.
- Interférences RF.
- Corrosion.
- Eau, feu, radiations.
- ...

L'environnement dans lequel opère le système embarqué n'est pas contrôlé ou contrôlable. Cela suppose donc de prendre en compte ce paramètre lors de sa conception. On doit par exemple prendre en compte les évolutions des caractéristiques électriques des composants en fonction de la température, des radiations...

Enfin pour terminer cette partie, les systèmes embarqués sont aujourd'hui fortement communicants. Cela est possible grâce aux puissances de calcul offertes par les processeurs pour l'embarqué (32 bits en particulier) et grâce aussi à l'explosion de l'usage la connectivité Internet ou connectivité IP.

La connectivité IP permet fondamentalement de contrôler à distance un système embarqué par Internet. Ce n'est en fait que l'aboutissement du contrôle à distance d'un système électronique par des liaisons de tout type : liaisons RS.232, RS.485, bus de terrain...

Cela permet l'emploi des technologies modernes du web pour ce contrôle à distance par l'utilisateur: il suffit d'embarquer un serveur web dans son équipement électronique pour pouvoir le contrôler ensuite à distance, de n'importe où, à l'aide d'un simple navigateur. Il n'y a plus d'IHM spécifique à concevoir pour cela, ce rôle étant rempli par le navigateur web.

Cela est une réalité : les chauffagistes proposent maintenant des chaudières pouvant être pilotées par le web !

Il faut aussi noter la montée en puissance des communications sans fil dans l'embarqué au détriment des communications filaires pour limiter le câblage et faciliter la mise en place du système embarqué. Le Wifi et toutes les normes de réseaux sans fil IEEE 802.15 comme Zigbee ont le vent en poupe dans l'embarqué et surtout en domotique (réseaux de capteurs sans fil par exemple).

Mais ne nous méprenons pas sur ces facilités et commodités, cela a bien sûr un revers : la sécurité du système embarqué puisque connecté à Internet.

La sécurité des systèmes embarqués est donc cruciale aujourd'hui et doit être prise en compte dès leur conception!

III.Les contraintes de temps et les systèmes embarqués

On entend souvent parler de Temps Réel dès que l'on parle de système embarqué.

En fait, un système embarqué doit généralement respecter des contraintes temporelles fortes (Hard Real Time) et l'on y trouve enfoui un système d'exploitation ou un noyau Temps Réel (Real Time Operating System, RTOS).

Le Temps Réel est un concept un peu vague et chacun a sa propre idée sur la question.

On pourrait le définir comme : Un système est dit Temps Réel lorsque l'information après acquisition et traitement reste encore pertinente.

Plus précisément, cela veut dire que dans le cas d'une information arrivant de façon périodique (sous forme d'une interruption périodique du système), les temps d'acquisition et de traitement doivent rester inférieurs à la période de rafraîchissement de cette information. Un temps maximum d'exécution est garanti et non un temps moyen.

Pour cela, il faut que le noyau ou le système Temps Réel :

- Soit déterministe : les mêmes causes produisent les mêmes effets avec les mêmes temps d'exécution.
- Soit préemptif : la tâche de plus forte priorité prête à être exécutée doit toujours avoir accès au processeur.

Ce sont là des conditions nécessaires mais malheureusement pas suffisantes pour affirmer qu'un système embarqué est Temps Réel par définition.

Une idée reçue est de mélanger Temps Réel et puissance de calcul du système embarqué. On entend souvent : Etre temps Réel, c'est avoir beaucoup de puissance : des MIPS (Million d'Instructions Par Seconde, des MFLOPS« opérations à virgule flottante par seconde » (en anglais,Mega FLoating point Operations Per Second).

Ce n'est pas toujours vrai. En fait, être Temps Réel dans l'exemple donné précédemment, c'est être capable d'acquitter l'interruption périodique (moyennant un temps de latence de traitement d'interruption imposé par le matériel), traiter l'information et le signaler au niveau utilisateur (réveil d'une tâche, libération d'un sémaphore…) dans un temps inférieur au temps entre deux interruptions périodiques consécutives.

On est donc lié à la contrainte durée entre deux interruptions. C'est donc bien le process extérieur à contrôler qui impose ses contraintes temporelles au système embarqué et non le contraire.

Si cette durée est de l'ordre de la seconde (pour le contrôle d'une réaction chimique par exemple), il ne sert à rien d'avoir un système à base de processeur 32 bits performant. Un simple processeur 8 bits voire même un processeur 4 bits fera amplement l'affaire ; ce qui permettra de minimiser les coûts sur des forts volumes de production.

Si ce temps est maintenant de quelques dizaines de microsecondes (pour le traitement des données issues de l'observation d'une réaction nucléaire par exemple), il est alors nécessaire de choisir un processeur nettement plus performant comme un processeur 32 bits (processeurs ARM, ColdFire..).

Dans le pire des cas, le traitement en Temps Réel sera réalisé en logique câblé tout simplement.

Il convient donc avant de concevoir le système embarqué de connaître la durée minimale entre 2 interruptions ; ce qui est assez difficile à estimer voire même impossible.

C'est pour cela que l'on a tendance à concevoir dans ce cas des systèmes performants et souvent surdimensionnés pour respecter des contraintes Temps Réel mal cernées à priori.

Ceci induit en cas de surdimensionnement un surcoût non négligeable.

IV.L'art de bien concevoir un système embarqué

Du point de vue technique, la conception d'un système embarqué demande à son concepteur d'être pluridisciplinaire : électronique, informatique, réseaux, sécurité ...

Mais le concepteur se doit aussi d'être un bon gestionnaire car concevoir un système embarqué revient finalement à un exercice d'optimisation : minimiser les coûts de production pour des fonctionnalités optimales.

Le système embarqué se doit d'être :

- Robuste.
- Simple.
- Fiable.

- Fonctionnel. Le système doit toujours fonctionner correctement.
- Sûr, surtout si la sécurité des personnes est en jeu.
- Tolérant aux fautes.

D'autres contraintes sont aussi à prendre en compte :

- L'encombrement.
- Le poids.
- Le packaging : difficulté de faire cohabiter dans un faible volume, électronique analogique, électronique numérique et RF sans interférences.
- L'environnement extérieur.
- La consommation électrique. Le système embarqué nomade doit être de faible consommation car il est alimenté par des batteries. Une consommation excessive augmente le prix de revient du système embarqué car il faut alors des batteries de plus forte capacité.
- Le coût. Beaucoup de systèmes embarqués sont fabriqués en grande série et doivent avoir des prix de revient extrêmement faibles.
- Le temps de développement. Dans un marché concurrentiel, il convient d'avoir un système opérationnel le plus rapidement possible pour être le premier sur le marché.

Devant toutes ces contraintes, le concepteur adopte des règles de bon sens :

- Faire simple.
- Utiliser ce que l'on a déjà fait ou fait par d'autres.
- Ne pas se jeter sur les technologies dernier cri. Quelle est leur pérennité dans le temps ?
- Ne pas se jeter sur le dernier composant sorti surtout s'il est grand public. Quelle est sa pérennité dans le temps surtout s'il l'on travaille pour la défense où l'on demande une maintenance sur 30 ans !
- Utiliser des technologies éprouvées qui ont fait leur preuve. Ces technologies peuvent d'ailleurs avoir plusieurs générations de retard par rapport à leurs homologues grand public.

Pour le grand public, le concepteur de systèmes embarqués peut sembler faire de l'inertie face aux nouvelles technologies mais il faut le comprendre : c'est pour le bien du système qu'il conçoit surtout si la sécurité des personnes est en jeu...

Cela explique en partie le décollage difficile des logiciels libres et de Linux pour l'embarqué.

V.Les logiciels libres et les systèmes embarqués

Les logiciels libres et en particulier GNU/Linux sont de plus en plus employés dans l'embarqué depuis qu'ils ont commencé à faire leur preuve il y a quelques années.

Regardons Linux. Pourquoi retrouve-t-on Linux dans l'embarqué ? Tout d'abord pour ses qualités qu'on lui reconnaît maintenant dans l'environnement grand public :

• C'est un logiciel libre disponible gratuitement au niveau source.

• Il est stable et efficace.

• Il n'y a pas de royalties à reverser sur chaque produit le mettant en œuvre.

• C'est un système d'exploitation ouvert.

• Différentes distributions sont disponibles pour coller au mieux à un type d'application.

• Il existe une aide rapide en cas de problèmes par la communauté Internet des développeurs Linux.

• Il y a un nombre de plus en plus important de logiciels libres disponibles.

• La connectivité IP chère aux systèmes embarqués est en standard.

Linux possède d'autres atouts très importants pour l'embarqué :

• Il existe un portage pour processeurs autres que la famille x86 : PowerPC, ARM, MIPS, 68K, ColdFire…

• La taille du noyau est modeste et compatible avec les tailles de mémoires utilisées dans un système embarqué (800 Ko pour µClinux sur processeur ColdFire).

• Différentes distributions spécialisées existent pour coller à un type d'application : routeur IP, PDA, téléphone…

• Le chargement dynamique de modules (drivers) est autorisé, ce qui permet d'optimiser la taille du noyau.

• La migration pour un spécialiste Linux à Linux embarqué est rapide et en douceur, ce qui réduit les temps de formation et les coûts…

On retrouve généralement aussi un certain nombre de suppressions de fonctionnalités dans les adaptations de Linux pour l'embarqué.

Il n'y a pas de gestion de la MMU (Memory Management Unit) pour ne pas pénaliser les performances globales du système. C'est le cas de µClinux, le noyau Linux adapté aux microcontrôleur sans MMU.

Les systèmes de fichiers en mémoire RAM (RAM Disk) ou en mémoire FLASH (JFFS2 : Journalling Flash File System 2) sont supportés.

On a en fait entendu parler pour la première fois officiellement de Linux pour l'embarqué à une exposition Linux World en 1999 où les sociétés Motorola, Force et Ziatech ont présenté un système CompactPCI fonctionnant sous «Linux embarqué».

En 2000 a été créé le consortium Linux embarqué (Embedded Linux Consortium) dont le but est de centraliser et de promouvoir les développements de solutions Linux embarqué.

Parallèlement s'est mis en place le portail « linuxdevices.com » dédié à Linux pour l'embarqué. Tout cela a contribué à la percée de Linux dans le monde de l'embarqué.

Qui dit systèmes embarqués dit souvent Temps Réel. Linux pour l'embarqué supporte aussi différentes extensions Temps Réel.

Embarquer Linux dans son système nécessite néanmoins des pré-requis concernant le matériel : le processeur doit être au moins un processeur 32 bits avec MMU (Linux) ou sans MMU (µClinux) couplé à quelques Mo de mémoire. Si vous êtes obligé d'utiliser un processeur 8 bits par exemple, il vous faudra vous orienter vers d'autres solutions logicielles...

Le tableau suivant montre l'usage de Linux embarqué sur processeurs 32 bit et plus.

Besoin	**Petit**	**Moyen**	**Haut de gamme**	**Embarqué haute disponibilité**
Taille RAM	< 4 Mo	2-8 Mo	8-32 Mo	> 100 Mo
TailleROM/FLASH	< 2 Mo	2-4 Mo	4-16 Mo	Go-To Disque Dur
Processeurs	DragonBall 68k ColdFire ARM	MIPS ColdFire ARM SH PowerPC x86		PowerPC x86
Caractéristiques matérielles	Pas de MMU	System on Chip SoC		
Exemples	PDA, téléphone	Routeur, Décodeur, imprimante		Commutateur téléphonique

tabeau 1.1 : Linux et le marché de l'embarqué 32 bits et plus

Enfin, Linux n'est rien sans tous les logiciels libres connexes et en premier lieu le compilateur GNU gcc, hautement configurable pour générer des compilateurs croisés.

Mais, il y a aussi tous les outils dédiés à l'embarqué, à faible empreinte mémoire offrant les mêmes fonctionnalités que leurs homologues pour Linux classique. On peut citer :

• L'outil BusyBox, véritable couteau suisse pour avoir toutes les commandes UNIX usuelles avec un seul exécutable.

- µClibc : la bibliothèque libc à faible empreinte mémoire.
- ...

VI. Conclusion

Nous voilà donc au terme de cette introduction sur les systèmes embarqués. Les points les plus importants vous ont été présentés afin d'aborder avec sérénité et sans à priori les articles suivants.

Quel avenir aux systèmes embarqués ? Un avenir radieux.

Quelles sont les évolutions techniques à venir ? Deux pistes semblent se dessiner :

- Un couplage fort entre matériel et logiciel via le développement conjoint matériel/logiciel ou codesign et l'approche système sur silicium SoC (System on Chip).
- L'explosion de l'électronique/informatique ambiante (ubiquitous computing) bon marché couplée à l'Internet ambiant. Les réseaux de capteurs (sans fil) seront omniprésents notamment en domotique.

Chapitre 2

Les Systèmes Programmables Sur Puce (PSOC)

Chapitre 2

Les Systèmes Programmables Sur Puce (PSOC)

I. Introduction Générale

PSoC : **P**rogrammable **S**ystem **O**n **C**hip, est une famille de circuits intégrés introduits au début des années 2000 par Cypress.

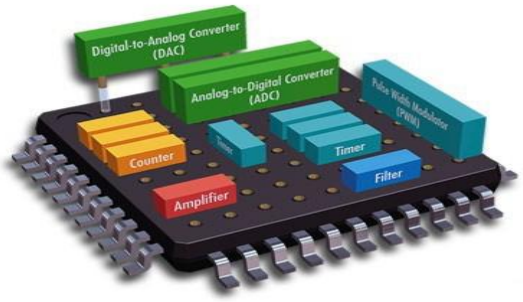


Figure 2.1 : schéma de PSoC

C'est un circuit intégré qui comprend un microcontrôleur et des fonctions logiques et analogiques configurables et interconnectables entre eux.

L'idée est de remplacer le microcontrôleur et les circuits d'interfaces analogiques (convertisseurs AN et NA, filtres, amplificateurs opérationnels, etc.) ou numériques (compteurs, timers, uart, interfaces pour bus divers, etc.) associés par un circuit unique. On intègre ainsi un système électronique embarqué complet dans un circuit intégré unique, ou tout au moins, on réduit très considérablement le nombre de composants.

Figure 2.2 : deux maquettes représentent avant et après le PSoC

Le PSoC est un circuit reconfigurable en fonctionnement : on peut par exemple imaginer un système embarqué qui s'arrête de fonctionner 1 à 2 minutes par jour pour se reconfigurer en modem et envoyer toutes les données qu'il à acquis dans la journée. Les fonctions utilisées pour la mesure sont reconfigurées en modem.

II. Architecture d'un PSoC

1. Définition :

PSoC est l'acronyme **P**rogrammable **S**ystem **O**n **C**hip, système électronique propriétaire de Cypress. Les circuits PSoC ont été introduits par Cypress au début des années 2000 et sont conçus pour remplacer à la fois le microcontrôleur et les circuits périphériques d'un système embarqué.

Comparé à un microcontrôleur 8 bits classique, le PSoC offre une architecture propriétaire Cypress de type Harvard, une horloge jusqu'à 24 Mhz, 4 Mips, une mémoire Flash jusqu'à 32 Ko, une mémoire SRAM (jusqu'à 2 K) pour les données, un bloc de multiplication 8x8 avec accumulation sur 32 bits.

Les PSoCs contiennent des blocs analogiques (majoritairement à capacités commutées) et numériques configurables par l'utilisateur permettant d'intégrer, entre autres :

- Des convertisseurs analogique/numérique et numérique/analogique;
- Des amplificateurs opérationnels et des amplificateurs d'instrumentation ;
- Des filtres et des comparateurs programmables ;
- Des compteurs et des timers ;
- Des UARTs supportant les communications RS232 full duplex ;
- Des contrôleurs de bus I2C et SPI;

Ce sont donc des Socs mixtes analogiques / numériques.

Une deuxième particularité est le caractère entièrement configurable de ces circuits, non seulement au niveau des fonctions du matériel qu'on y implante que du type de signal affecté à chacune des broches du circuit intégré (entrée, sortie, analogique, numérique, …)

Des bibliothèques de modules utilisateurs préconfigurés permettent d'implanter dans les PSoCs, grâce à l'environnement de développement intégré PSoC Designer™ (disponible gratuitement), les fonctions analogiques, numériques ou mixtes, simples ou complexes précitées. Cette implantation, ainsi que celle des programmes développés, se fait in-situ par l'utilisateur, dans la mémoire Flash.

Les programmes interagissent avec les modules utilisateurs hardware, qu'ils contrôlent par l'intermédiaire d'APIs (**A**pplication **P**rogramming **I**nterfaces) générés automatiquement à la fin de la phase de configuration matérielle.

PSoC Designer™ permet aussi l'écriture et la mise au point des programmes (en assembleur ou en C).

La troisième originalité des PSoCs est leur reconfigurabilité dynamique. En effet, les informations de configuration, contenues dans la mémoire Flash, sont chargées dans des registres SRAM à la mise

sous tension. Ces registres sont modifiables par le programme applicatif qui peut y écrire directement .Cette reconfiguration va du simple changement de la valeur du gain d'un amplificateur à la modification du type des fonctions implantées dans les blocs hardware et du type de signal affecté aux broches du circuit intégré. Cette possibilité trouve de nombreuses applications dans le domaine de la mise en forme des signaux issus de capteurs.

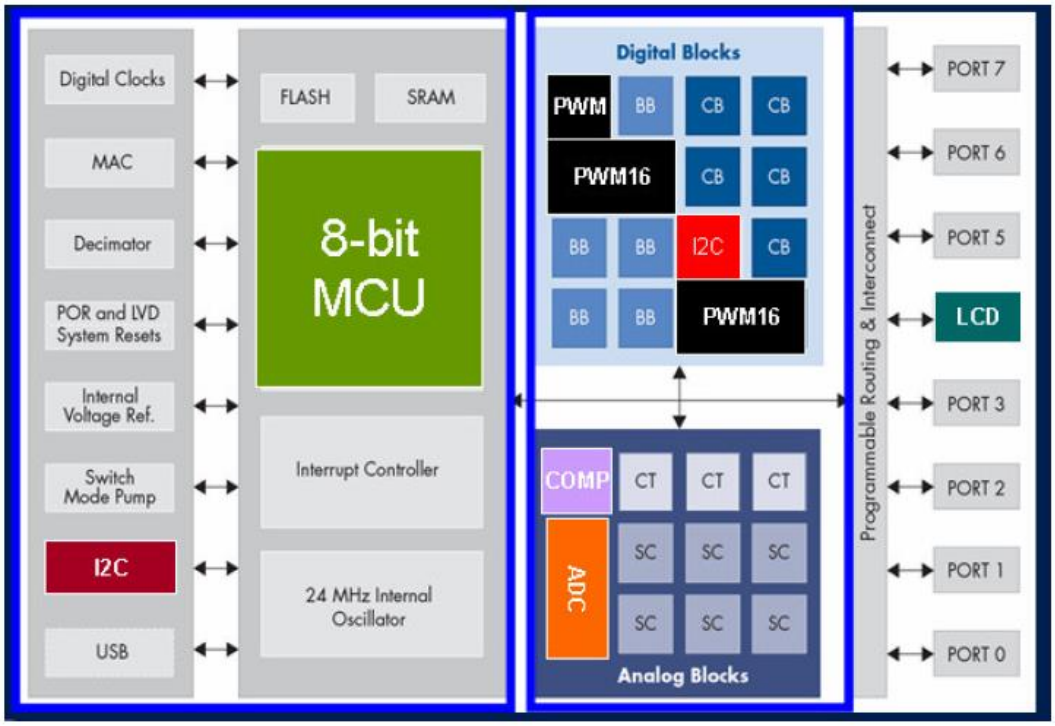


Figure 2.3 : Architecteur de PSoC

2. Les blocs numériques et analogiques :

- les blocs numériques et analogiques banalisés configurables intègres : UART, SPI, timer, PWM, amplificateur op, filtres, convertisseurs, etc.
- Un seul modèle peut répondre à une multitude d'applications le microcontrôleur ne se retrouve pas encombré de fonctions superflues.
- Les blocs sont configurables et reconfigurables en fonctionnement, permettant d'adapter, de modifier, d'optimiser dynamiquement les principaux fonctions réalisées.

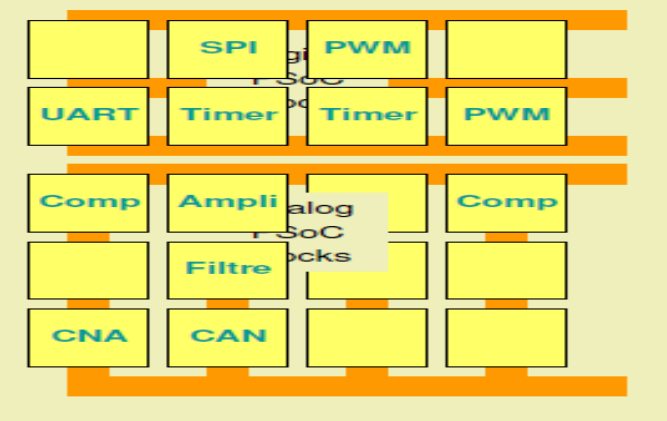


Figure 2.4 : Schéma des blocs analogiques et logiques

2.1. Blocs analogiques

- 12 Blocks analogiques (4 colonnes)
- 2 types:
  - ✓ CT : Continuous Time
    - fonctions simples : amplificateur op, PGA, comparateur
  - ✓ SC: Switcher Capacitors
    - fonctions complexes : filtrage, conversion, etc.
- Cascadables et interconnectables

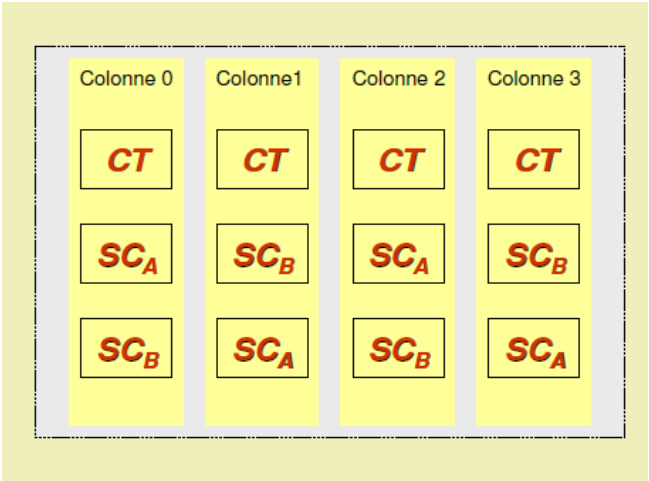


Figure 2.5: Schéma des Blocs analogiques

2.2. Blocs numériques :

Il y a 8 Blocks numériques, chacun constituant une 'tranche' de 8 bits

- Chaque bloc est constitué de 3 registres Data et 4 registres de configuration.
- Cascadables et interconnectables.

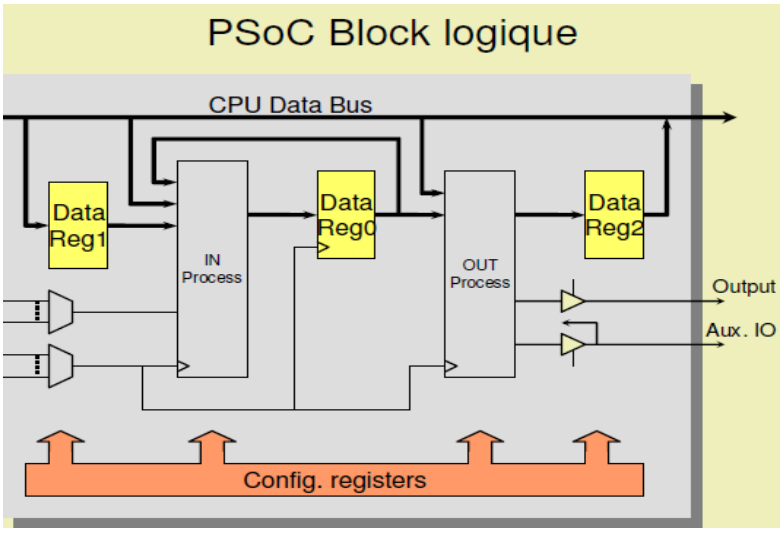


Figure 2.6: Schéma de Block logique

3. Mémoire Flash :

La mémoire flash est une mémoire de masse à semi-conducteurs réinscriptible, c'est-à-dire une mémoire possédant les caractéristiques d'une mémoire vive mais dont les données ne disparaissent pas lors d'une mise hors tension. Ainsi, la mémoire flash stocke les bits de données dans des cellules de mémoire, mais les données sont conservées en mémoire lorsque l'alimentation électrique est coupée.

Sa vitesse élevée, sa durée de vie et sa faible consommation (qui est même nulle au repos) la rendent très utile pour de nombreuses applications : appareils photo numériques, téléphones cellulaires, imprimantes, assistants personnels (PDA), ordinateurs portables ou dispositifs de lecture et d'enregistrement sonore comme les baladeurs numériques, clés USB. De plus, ce type de mémoire ne possède pas d'éléments mécaniques, ce qui lui confère une grande résistance aux chocs.

La mémoire flash est un type d'EEPROM qui permet la modification de plusieurs espaces mémoires en une seule opération. La mémoire flash est donc plus rapide lorsque le système doit écrire à plusieurs endroits en même temps.

Elle contienne :

- 4K à 16Koct. selon le composant.
- Process Cypress SONOS (Flash).
- organisation par blocs 64 octets.
- 50.000 cycles éffac/prog par bloc.
- Programmation à tension nominale.

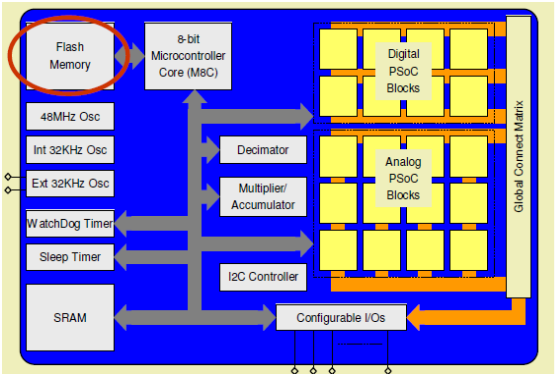


Figure 2.7 : Image de la mémoire Flash

4. Mémoire SRAM :

La mémoire vive (SRAM) est généralement définie en opposition à la mémoire morte (ROM) : les données contenues dans la mémoire vive sont perdues lorsque l'alimentation électrique est coupée alors que la mémoire morte conserve ses données en absence d'alimentation électrique. La mémoire morte n'est donc pas volatile, ce qui la rend nécessaire lors du démarrage d'un ordinateur. En effet, la mémoire vive est dans un état indéterminé lors du démarrage.

Le sens littéral des termes SRAM et mémoire vive peut prêter à confusion. En effet, le terme SRAM implique la possibilité d'un accès arbitraire aux données, c'est-à-dire un accès à n'importe quelle donnée n'importe quand, par opposition à un accès séquentiel, comme l'accès à une bande magnétique, où les données sont nécessairement lues dans un ordre défini à l'avance.

La SRAM présente la particularité de pouvoir être accédée à la fois en lecture et en écriture. Une activation électronique appropriée permet si besoin de verrouiller temporairement en écriture des blocs physiques donnés.

Dans SRAM d'un PSOC il y a :

- Mémoire paginée (numéro de page dans le registre Flag)
- Jusqu'à 8 pages de 256 octets
- 256 octets pour les premiers composants

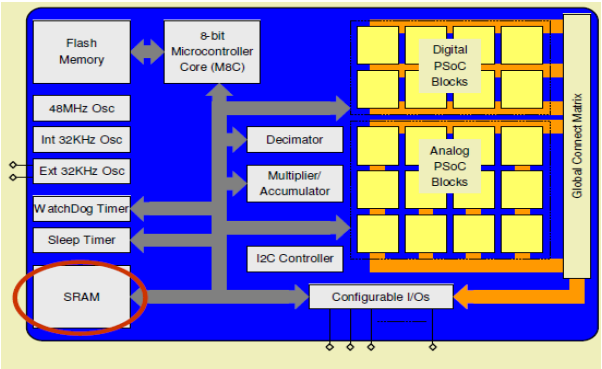


Figure 2.8 : Image de mémoire SRAM

5. Modèle de Programmation :

L'architecteur de Programmation est comme suit :

- Pointeur de Pile SP (8 bits)
- Accumulateur A (8 bits)
- Compteur Ordinal PC (16 bits)
- Index X (8 bits)
- Flags F (8 bits)

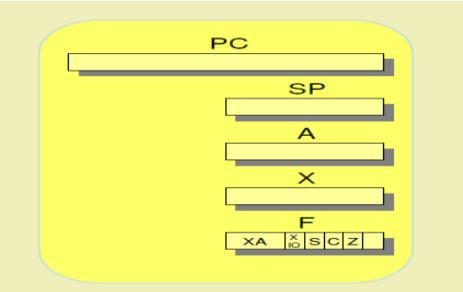


Figure 2.9 : diagramme de Modèle de Programmation :

III. Représentation de la maquette de développement de PSoC «CY8CKIT-001»

1. Présentation de la maquette

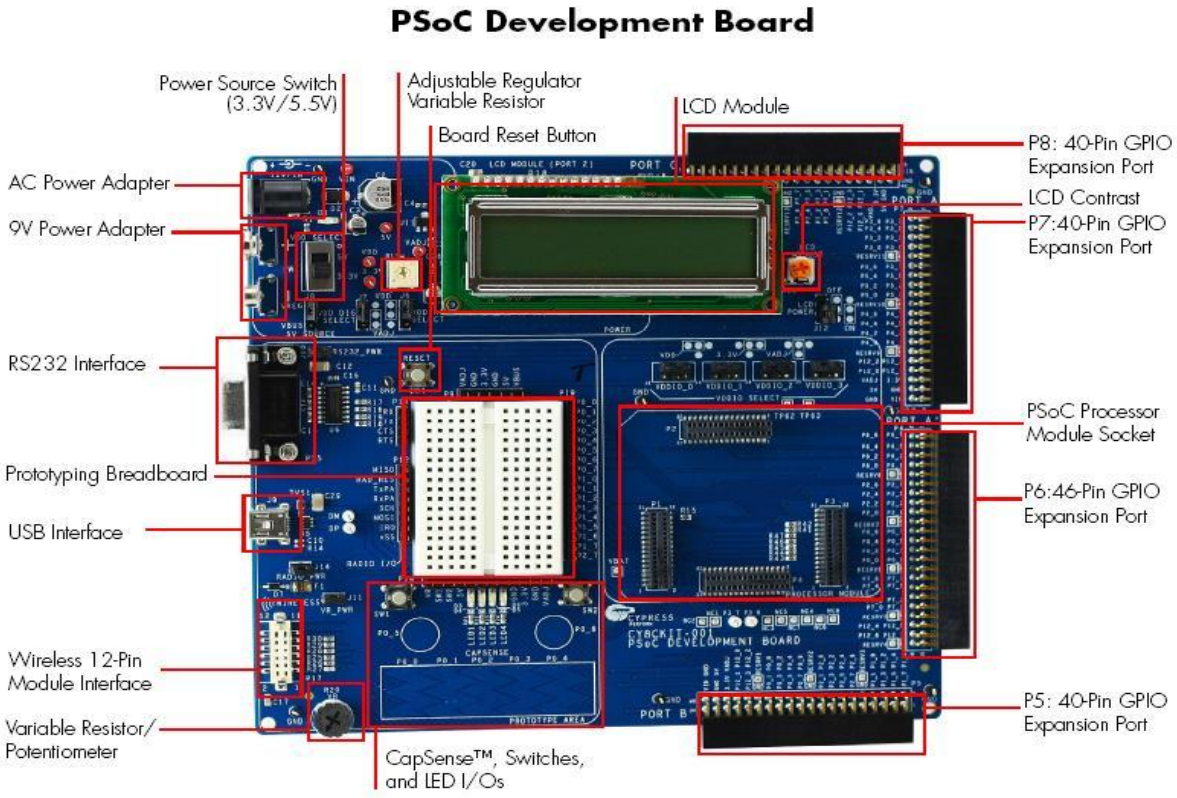


Figure 2.10 : Image de kit de développement PSoC

Le kit comprend :

- ✓ une carte d'évaluation incluant un support pour un PSoC DIL 28 broches, 1 afficheur LCD, 4 LEDs, 1 potentiomètre, circuit d'alimentation, une interface RS232, une connecteur pour programmation, une espace « Labdec » pour prototypage(mini plaque a essai).
- ✓ 4 circuits PSoC (CY8C29466-24PXI [PSOC1] , CY8C38446-24PXI [PSOC3], CY8C4245AXI-483 44TQFP [PSOC4] et CY8C5568AXI-060 TQFP100 [psoc5])
- ✓ Programmateur MiniProg + câble USB
- ✓ jeu de fils pour faire des connexions CD et notice sur la carte d'évaluation, les ports du PSoC sont accessibles sur des connecteurs.

Figure 2.11 : Image de composant de maquette

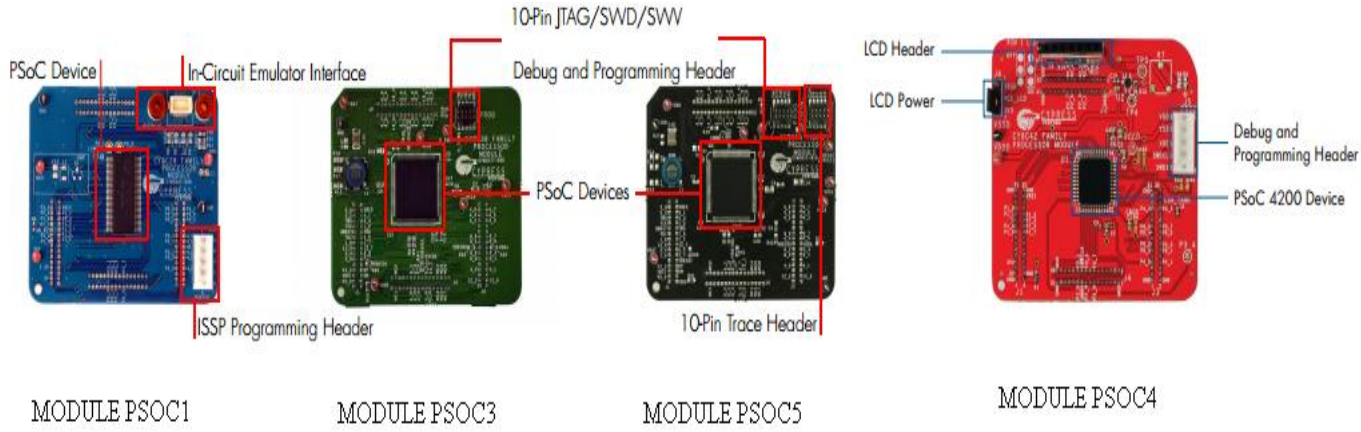


Figure 2.12: Image représente les quatre familles de PSoC

En PSoC Cypress il y 3 Familles PSoC1, PSoC3, PSoc5, Dans notre projet nous avons exploite le trois

PSoC1 : PSoC CY8C29 processeur MC8 (4 MIPS)

PSoC3 : PSoC CY8C38 processeur 805L (33 MIPS)

PSoC4 : PSoC CY8C42 processeur ARM-Cortex0 (43 DMIPS)

PSoC5 : PSoC CY8C55 processeur ARM-Cortex3 (100 DMIPS)

2. Caractéristiques des PSoC

	PSOC1	PSOC3	PSOC4	PSOC5
Processeur	M8C	8051	ARM-Cortex-M0	ARM-Cortex-M3
Vitesse	24 MHz pour 4 MIPS	67 MHz pour 33 MIPS	48MHz CPU pour 43 DMIPS	67 MHz pour 100 DMIPS
Taille du bus de données	8 bits	8 bits	32 bits	32 bits
Mémoire programme	Flash de 4 ko à 32 ko.	Flash de 8 ko à 64 ko	Flash de 32 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 4 ko	SRAM de 16 ko à 64 ko
Alimentation	1,7 V à 5,25 V.	0,5 V à 5,5 V	de 1,71 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	Actif : 6 mA hibernation : 300 nA
Conversion A/N	1 Delta-Sigma de 14 bits	1 Delta-Sigma de 20 bits	1 SAR de 12 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	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	USB 2.0, I2C, SPI, UART, LIN, I2S
Entrées / sorties	64	72	36	72

Tableau 2.1 : les caractéristiques de 3 familles de PSoC

IV. Environnement de développement de PSoC

1. Définition

Cypress fournit des outils de développements gratuits téléchargeables gratuitement sur son site

Figure 2.13 : Schéma des outils de développement de PSoC

Le développement des PsoC1 s'effectue à l'aide du logiciel **PSoC Designer** (Version 5.4 à ce jour, février 2014, incluant le compilateur C gratuit).

Le développement des PSoC 3 et 5 se fait à l'aide du logiciel **PSoC Creator**(Version 2.2 ou 3.0 à ce jour, février 2014, incluant le compilateur GNU C/C++ gratuit ou compilateur CA51 de KEIL).

La programmation du circuit peut se faire en suie à l'aide de PSoC programmer, accessible aussi à partir de PSoC Designer et PSoC Creator. Il est nécessaire de disposer d'un petit accessoire de programmation (périphérique USB) type PSoC MiniProg.

2. Les types de PSoC

A ce jour (2014), il existe 4 grandes familles de PSoC, nommées PsoC1, PSoC 3 , PSoC 4 et PSoC 5.

PSoC 1 : C'est la famille d'origine qui date du début des années 2000.

PSoC 3 et PSoC 5 : Deux familles de PSoC introduites par Cypress début 2010.

PSoC 4 : Une nouvelle famille de PSoC introduite par Cypress début 2013.

3. PSOC Designer

PSoC Designer présente une page d'accueil. Pour commencer un projet, cliquer sur **File** (en haut à gauche) puis **New Project**.

Choisir un nom et un répertoire de travail, puis (écran suivant) choisir un circuit (**device**), pour le kit d'évaluation, le circuit à utiliser est le **CY8C29466-24PXI**. Choisir un langage de programmation le langage assembleur ou le langage C.

On accède ensuite à l'écran principal de PSoC Designer qui est divisé en plusieurs fenêtres

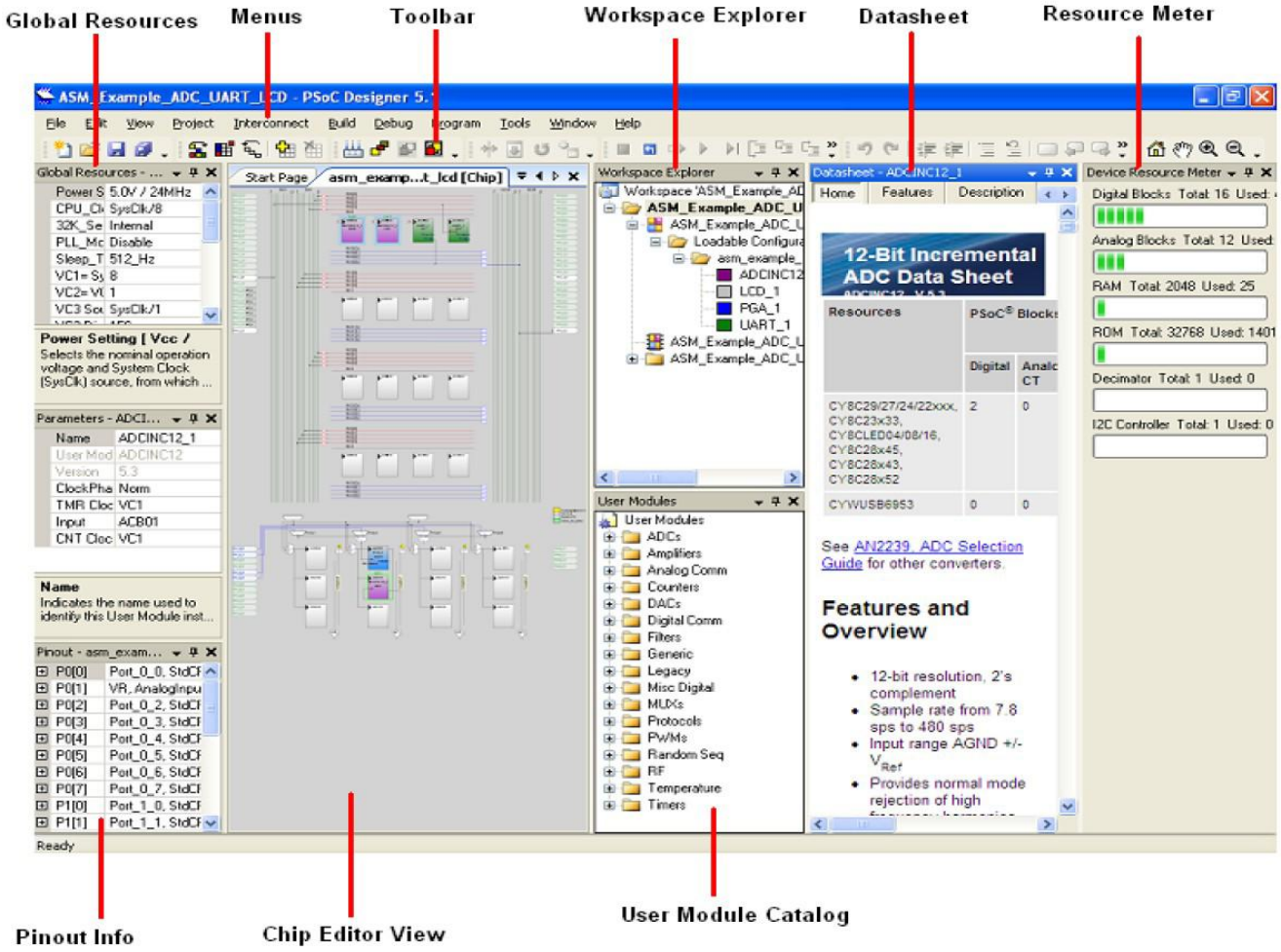


Figure 2.14 : Image représente la primière fenêtre de PSoC Designer

Il faut mentionné que la conception d'après Cypress se fait en 4 étapes.

Conception en 4 étapes	1. Sélection et placement des modules	2. Configuration des modules et des drivers d'entrées et sorties	3. Connexion des modules	4. Coder, Compiler vérifier, débugger
	Choisir les modules analogiques et numériques	Configurer les paramètres globaux (horloge, etc..) ; les parametres des modules et les entrées sorties (type, vitesse, etc..)	Routage des connections avec l'outil graphique.	Programmation en C ou en assembleur, compilation et programmation.

Remarque : Les 4 étapes décrites ici ne sont ni formelles, ni irréversibles, rien n'empêche par exemple de rajouter un module après avoir des lignes de code

3.1. Sélection et placement des modules

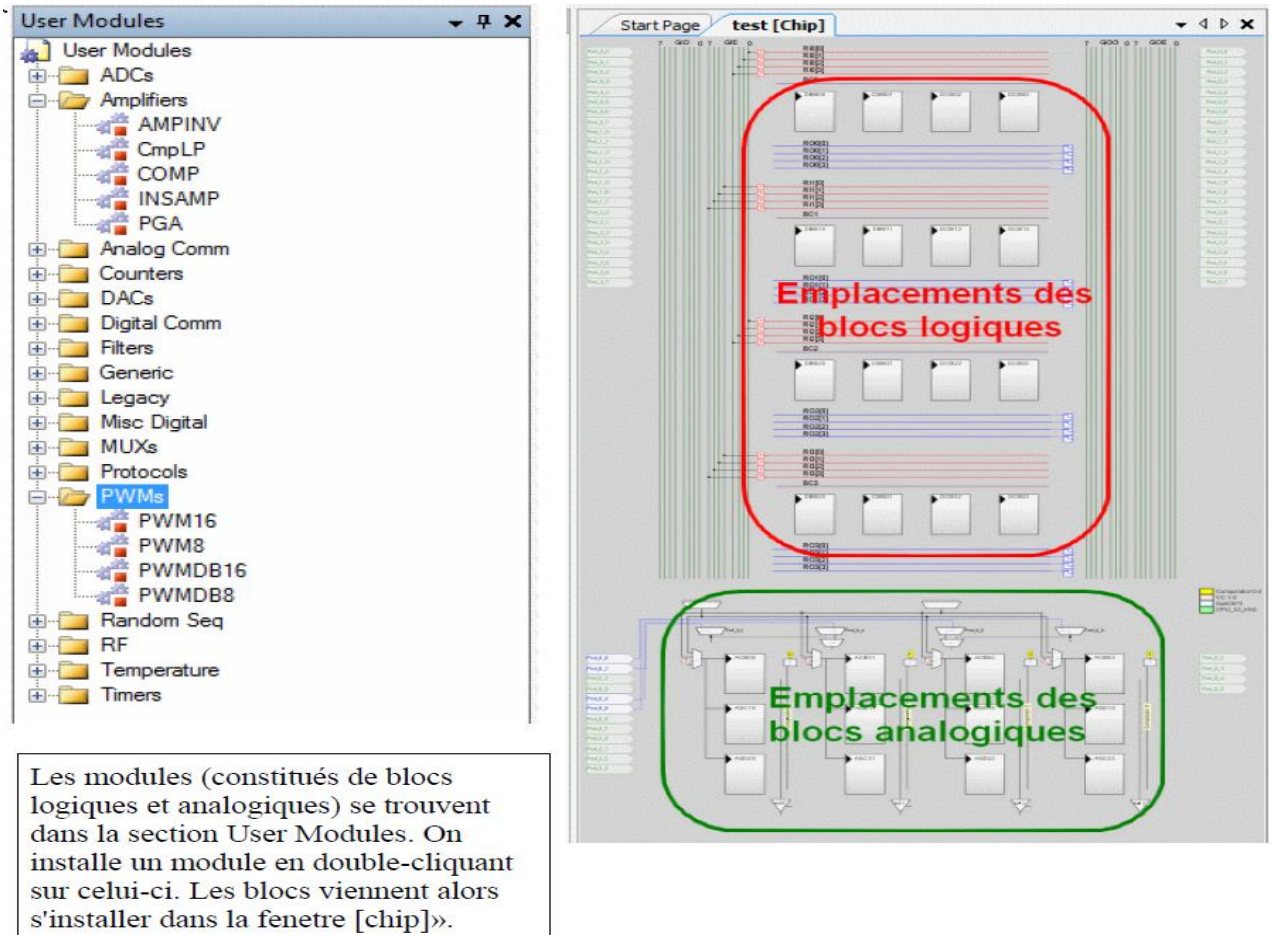


Figure 2.15 : Schéma représente les emplacements de Blocs Logiques et analogiques

Il est possible d'accéder à la documentation de chaque module par un clic droit sur le module puis Datasheet.

Attention : un seul module peut occuper plusieurs blocs, logiques ou analogiques (en fonction de la complexité). Exemple : 1 module PWM 8 bits occupe un bloc logique, un module PWM 16 bits occupe 2 blocs logiques, un module amplificateur inverseur occupe un bloc analogique, un module ADCIN (CAN incrémental) occupe un bloc logique et un bloc analogique.

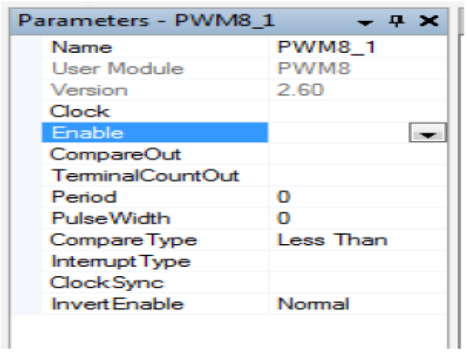
Les blocs analogiques ne peuvent pas être placés n'importe ou : la première ligne de modules analogiques reçoit les blocs correspondant à des modules amplificateur inverseur ou non inverseur tandis que les deux lignes suivantes de modules reçoivent des blocs à capacités commutés (filtres, etc..).

Les modules installés dans la fenêtre [chip] peuvent être déplacés à la souris par glissé-déposé.

3.2. Configuration des modules et des drivers d'entrées et sorties

Au cours de cette étape, il faut configurer et paramétrer les différents modules, les paramètres globaux et les entrées et sorties du circuit.

Configuration des modules



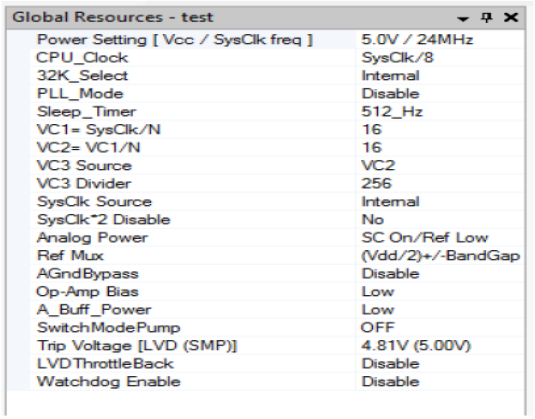
La fenêtre Paramètres (Parameters) permet de configurer chaque module utilisé.

Ici, pour un PWM 8 bits, on peut configurer les entrées horloge et Enable, les sorties, la période, la largeur etc..

Toutes ces données peuvent aussi être modifiées par le programme C. Les instructions détaillées se trouvent dans la « datasheet » du composant accessible par le logiciel.

Figure 2.16 : Les configurations des modules d'entrées et de sorties

Configuration des paramètres globaux



La fenêtre Ressources globales (Global Ressources) permet de configurer les paramètres du PsoC communs aux différents modules placés, en particulier les circuits d'horloge VC1, VC2, VC3 (VC : Variable Clock), diviseurs programmables qui peuvent attaquer les circuits compteurs ou PWM.

Figure 2.17 : Représente la configuration des paramètres globaux

Configurations des entrées et sorties

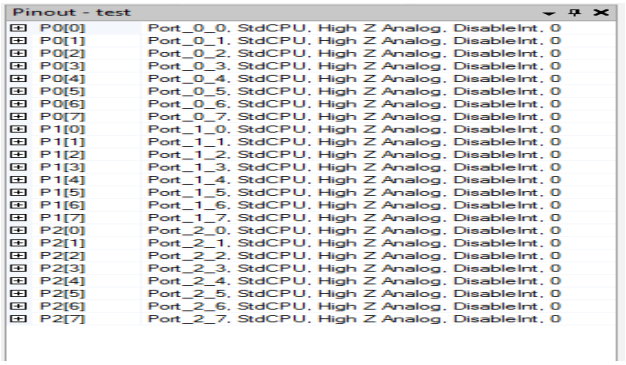


Figure 2.18 : Représente la configuration d'entres et de sorties

La fonction de chacune des broches d'entrée et sortie du PSoCs est paramétrable :

- Entrée et/ou Sortie standard (accessible en lecture et/ou en écriture dans l'espace registre)
- Entrée et/ou Sortie globale (connexion aux PSoC Blocks numériques)
- Entrée et/ou Sortie analogique.

L'interface d'entrée ou de sortie est paramétrable :

• Strong (CMOS), Pull-up, Pull-down, High-Z, Open Drain

Le Mode Interruption est configurable pour chaque broche :

• Front montant, descendant, ou changement d'état

La configuration des broches d'entrées/sorties se fait soit dans la fenêtre Pinout, soit dans la fenêtre [chip] (dessin du circuit) en cliquant sur chacune des broches.

3.3. Connexion des modules

L'interconnexion des modules se fait graphiquement dans la fenêtre [chip]. Les connexions se font à l'aide de multiplexeurs que l'on configure en cliquant dessus. Tout se fait graphiquement.

Attention n'importe quel module placé ne peut pas être relié à n'importe quelle entrée : d'où l'importance du placement

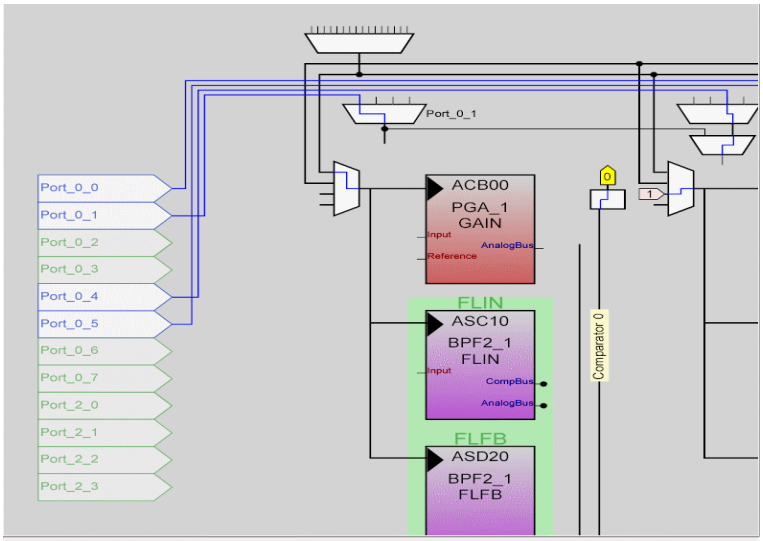


Figure 2.19: Interconnexion des modules analogiques

Pour les modules numériques (voir figure suivante) on relie les sorties à des colonnes sur lesquelles pourront se connecter les sorties.

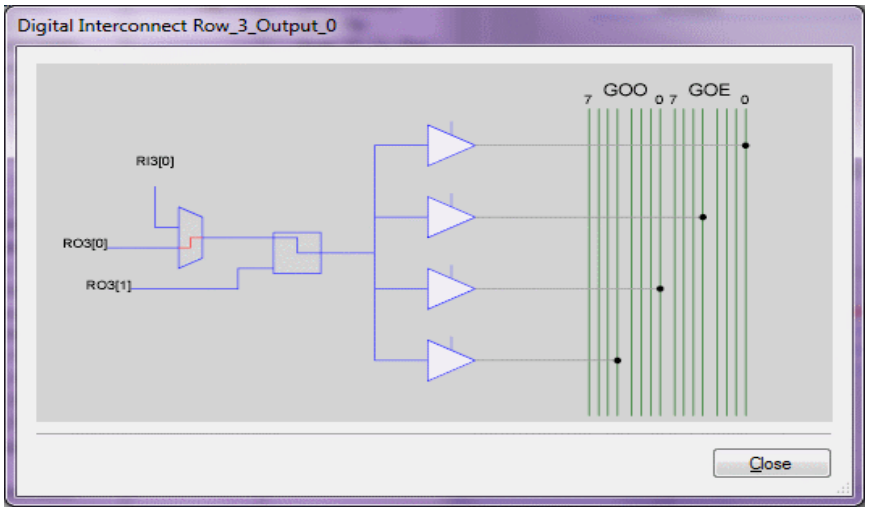


Figure 2.20 : Interconnexion des modules analogiques

3.4. Coder, compilé et Programmé

Il reste maintenant à coder en C. Pour cela, il faut ouvrir le programme principal avec la fonction File puis Open File. Le programme principal est crée par défaut par PSoC designer, il est situé à la racine du projet et s'appelle main.c.

Quand on débute un projet, celui-ci est vide, à l'exception de l'en-tête suivant, et est prêt à être complété

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------
#include <m8c.h> // Bibliothèque des constantes et les macros spécifiques
#include "PSoCAPI.h"  // définitions des l'API PSoC pour tous les modules de
                      //l'utilisateur
Void main (void)
{
/ / Insertion du code de routine principale ici.
}
```

Pour bien utiliser les instructions C relatives à l'utilisation d'un module, se reporter au paragraphe Sample code (échantillon de programme) de la datasheet.

Le reste est de la programmation en C « standard ».

Pour compiler, vérifier et générer les fichiers de configuration du PSoC, il suffit de cliquer sur l'icone Build :

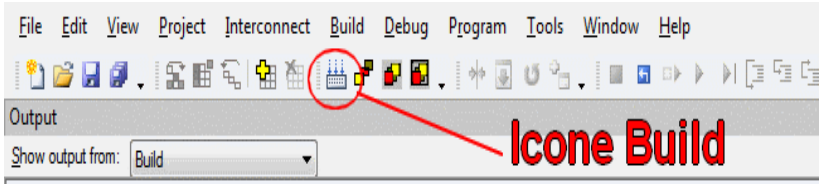


Figure 2.21: Schéma représente l'icône Build

Enfin, il ne reste plus qu'à programmer le PSoC en cliquant sur le menu Program qui ouvre la fenêtre de programmation

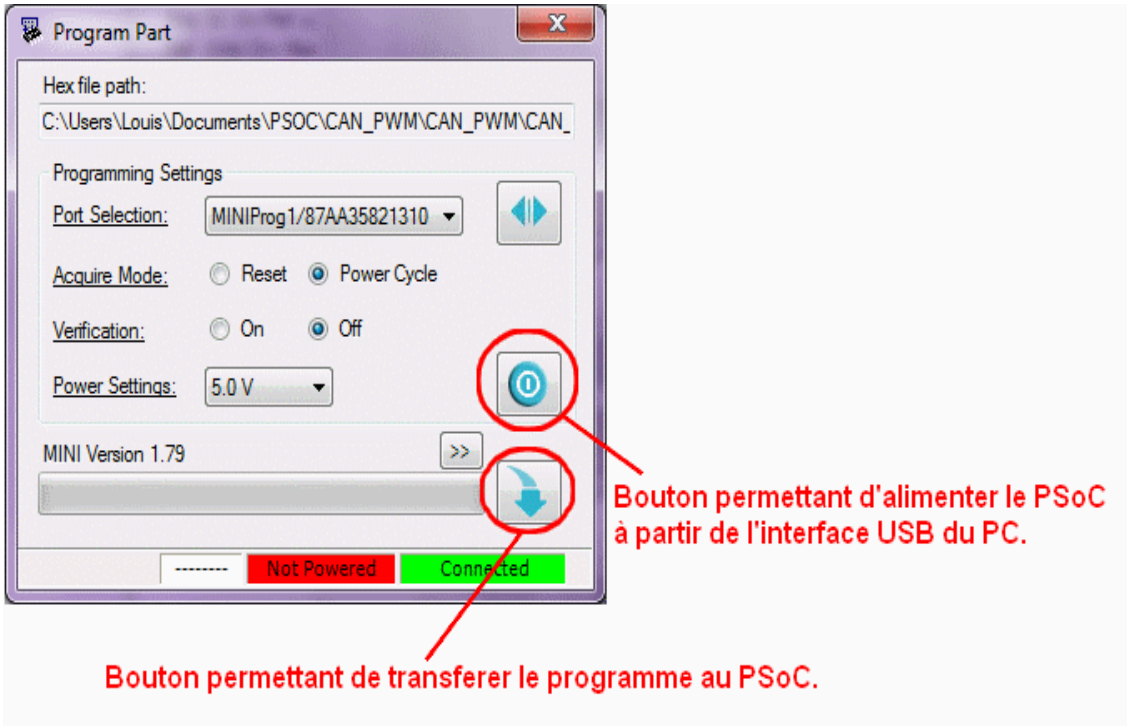


Figure 2.22 : Programmer PSoC

On transfère dans un premier temps le programme dans le PSoC, ensuite, il est possible de faire fonctionner la carte d'évaluation en l'alimentant via l'interface USB du PC

4. PSoC Creator

Après avoir installe PSoC Creator. Pour commencer un projet, clique sur File (en haut à gauche) puis Empty PSoC Design ensuit choisir un nom et un répertoire de travail.

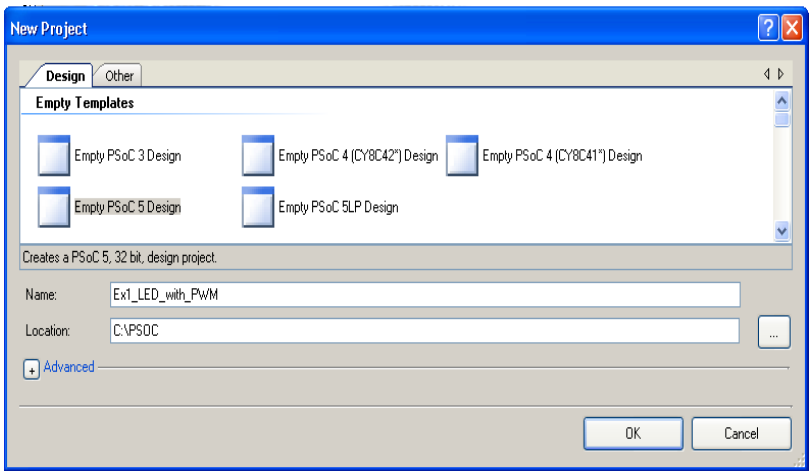


Figure 2.23 : Primaire fenêtre de PSoC Creator

La conception de projet avec PSoC Creator passe par 4 étapes :

4.1. Mise en place et configure les composants :

Ce logiciel contient 56 composants analogiques, CapSense, Communications, Digital, Filtres et System se trouvent dans la section Cypress.

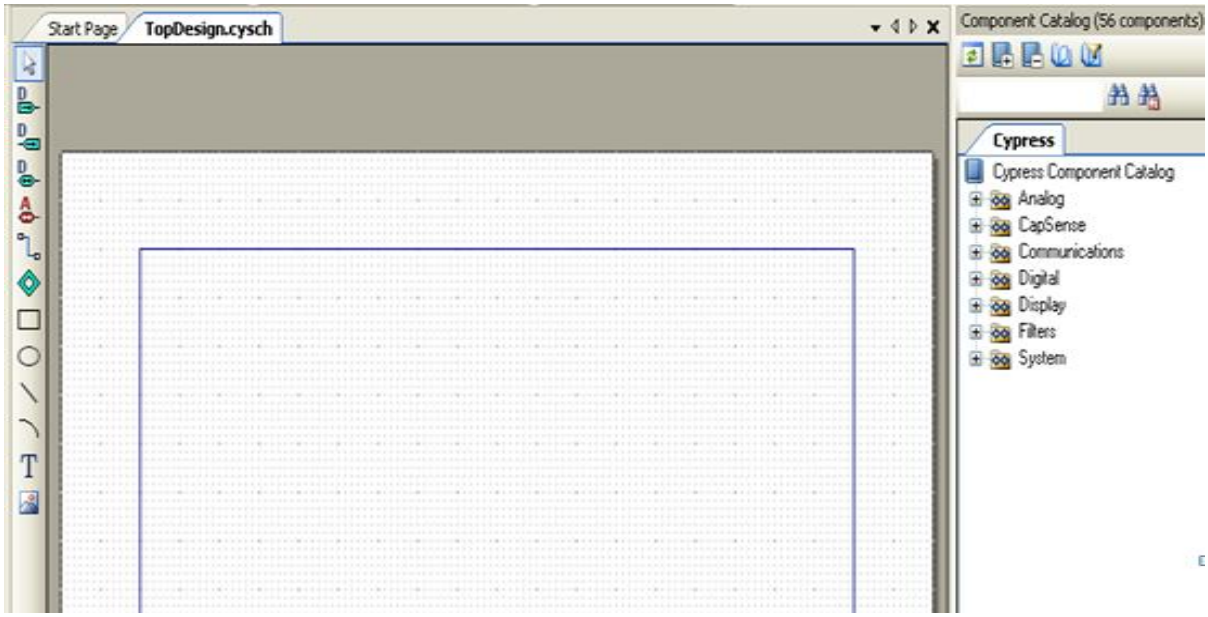


Figure 2.24 : Représente les nombres de composants de PSoC Creator

Pour installer un bloc il faut cliquer sur le bouton droit et entré. Puis double clique sur celui-ci pour modifier leur paramètre. Après règler le tableau de Pin

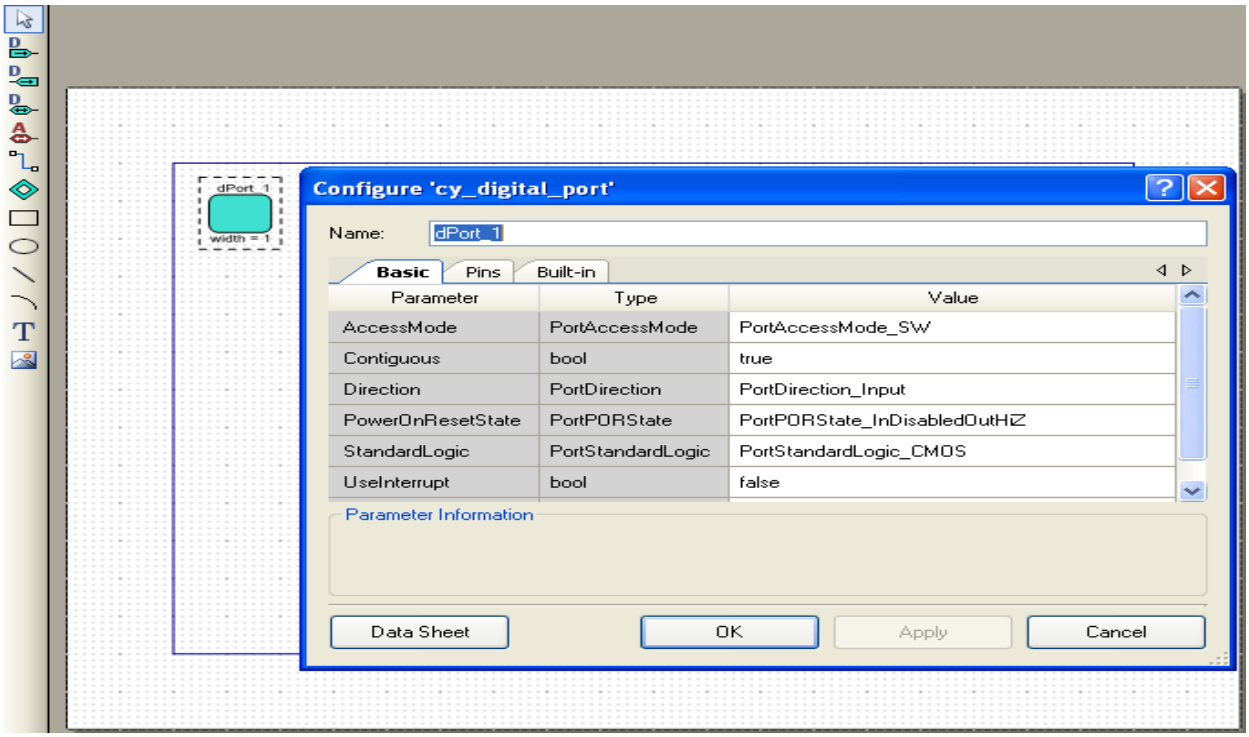


Figure 2.25: Représente la déclaration de composant

Il est possible de consulter des informations sur chaque bloc par un clic sur Datasheet.

Par exemple mettre en place et configurée une LED

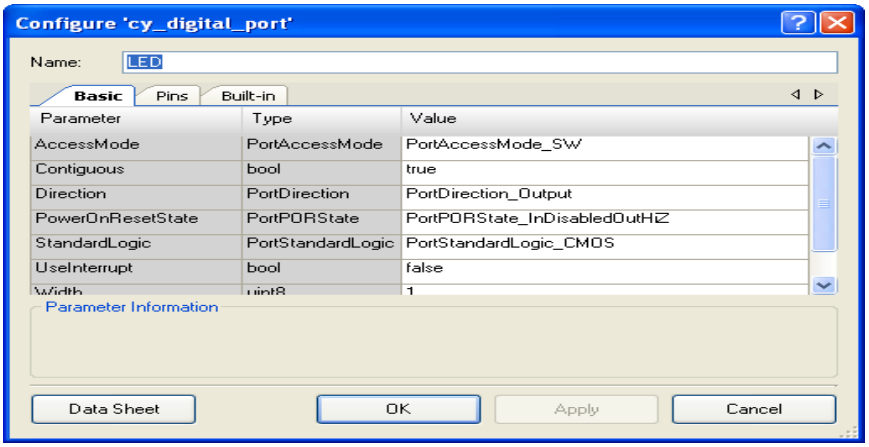


Figure 2.26: Configuration de LED

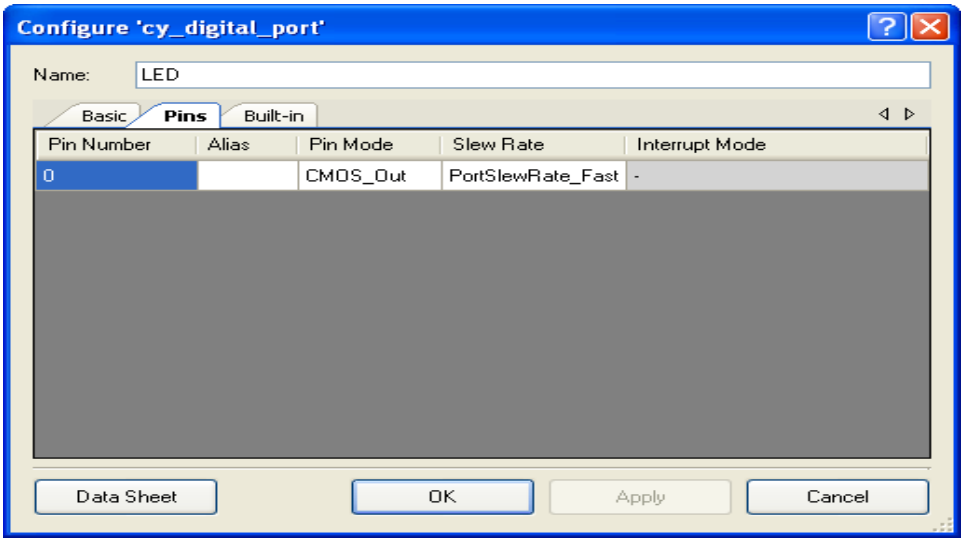


Figure 2.27 : Pins de LED

4.2. Connexion des composants :

La connexion des composants en PSoC Creator se fait par 4 méthodes :

A/ L'outil Wire Tool : Cette fonction se trouve sur l'accueil de logiciel

Figure 2.28 : Icône de Wire Tool

B/ Composant Digital logic

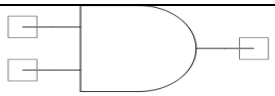
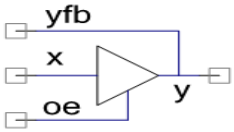
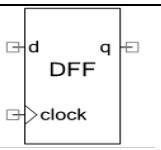
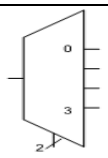
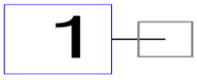
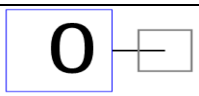
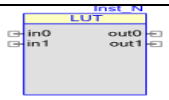
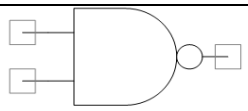
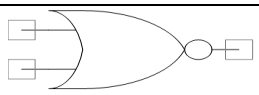
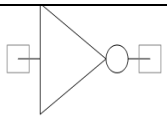
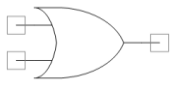
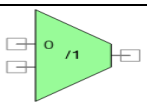
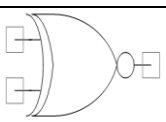
Les Fonctions	Les symboles
And	
Bufoe	
D Flip Flop	
Multiplexeur	
Logic High	
Logic Low	
Lookup Table	
Nand	
Nor	
Not	
Or	
Virtual Mux	
Xnor	
Xor	

Tableau 2.2 : Les fonctions et les symboles de digitales logique

C/ Composant System

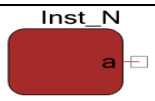
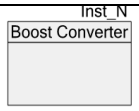
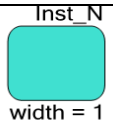
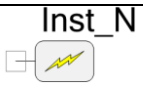
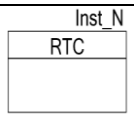
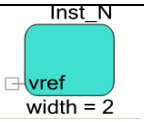
Les Fonctions	Les Symboles
Analog Port	
Boost Converter	
Clock	
Die Température	
Digital Port	
DMA	
EEPROM	
Interrupt	
RTC	
SIO Port	
Vref	

Tableau 2.3: les fonctions et les symboles de composant System

4.3. Configuration de Pins :

Pour réaliser cette étape, il faut tout d'abord entrer dans Workpace Explore puis double cliquer sur le fichier qui contient le nom projet.cydwr. Ensuite cliquer sur Pins. Enfin sélectionner les Pins convenables.

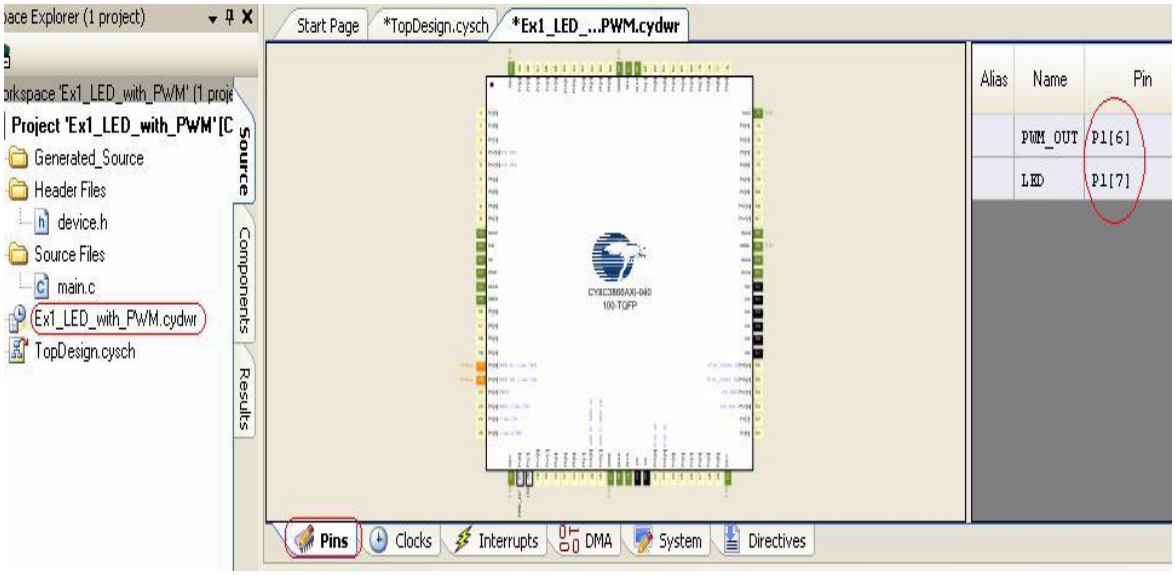


Figure 2.29 : Schéma représente la configuration de Pins

Cette étape se fait pour choisir les ports de sortie.

4.4. Création, Compiler, Vérifier et Transférer le programme :

Il faut maintenant écrire le programme en C pour cela il faut ouvrier le programme principal avec la fonction **main.c**. la programme est crée par défaut en PSoC désigner.

```
#include <device.h>
#define MS_DELAY 167u /* For delay, about 167ms */
/*****************************************************************************Nom de Fonction: main
*****************************************************************************
*Résumé:
* La fonction principale initialise le PWM et PWM commence l'horloge, qui
* Clignote LED1 à environ une fois par seconde. Puis la boucle principale est entré
* Ce qui retarde assez pour LED2 à clignoter à un rythme plus rapide que LED1
#include "PSoCAPI.h"  // définitions des l'API PSoC pour tous les modules de
                                    //l'utilisateur
Void main (void)
{
/ / Insertion du code de routine principale ici.
}
```

Pour compiler et Générer les fichiers de configuration du PSoC, il suffit de cliquer sur l'icône Build

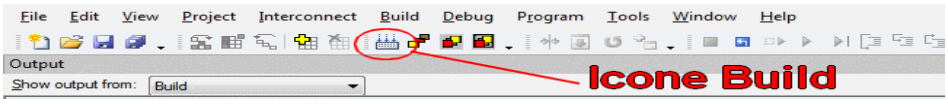


Figure 35 : Icône Build

Enfin, il ne reste plus qu'à programmer le PSoC en cliquant sur menu **Tools** PSoC Creator La fenêtre Options s'ouvre **:**

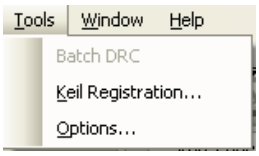


Figure 2.30 : Menu Tools de PSoC Creator

Dans la fenêtre **Options**, Sélectionnez **Programmer/Debugging →MiniProg3** dans la liste.

- Réglez **Applied Voltage** à **3.3V**
- Réglez **Transfer Mode** à **SWD**
- Réglez **Active Port** à **10 Pin**
- Réglez **Acquire Mode** à **Reset**
- Réglez **Debug Clock Speed** à **3.2 MHz**
- Clic sur **OK**.

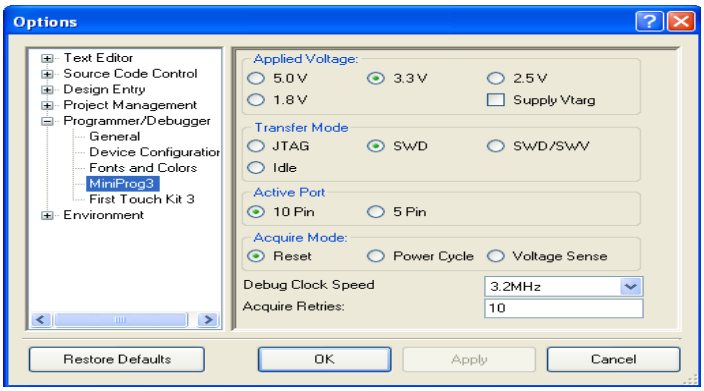


Figure 2.31 : Menu de Réglage pour Programmé v le Microcontrôleur par Creator

Dans le menu **Debug**

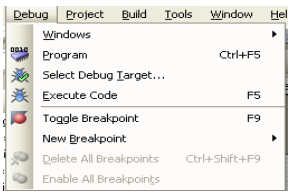


Figure 2.32 : Menu Debug de dialogue

Cliquer sur **Select Debug target.** La boite de dialogue Sélectionnée s'ouvre

Développez l'arborescence sous **MiniProg3** et cliquez sur **Port Acquire**

Sélectionnez le périphérique approprié et cliquez sur **Connect**

Cliquer sur **close**

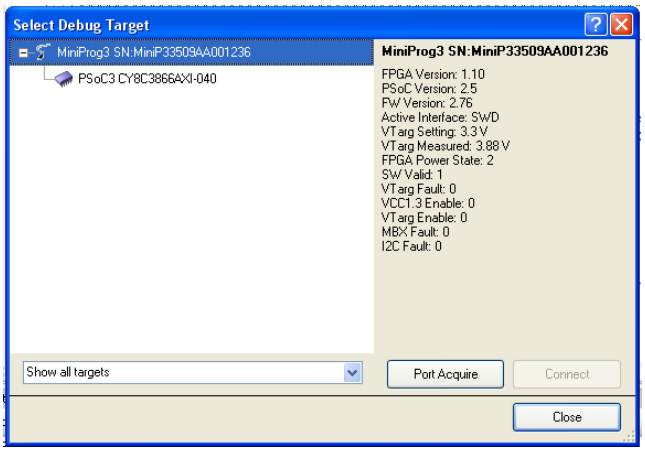


Figure 2.33: Menu de Connection

On transfère dans un premier temps le programme dans le PSoC après il est possible de faire fonctionner la carte d'évaluation en l'alimentant via l'interface USB du PC ou via l'alimentation externe du secteur (tranfort 5 v = ou via une pile 9v.

5. Des applications avec PSoC :

Elles sont très variées ainsi que le montre ce document Cypress.

Figure 2.34 : Images des applications avec PSoC

6. Les différences entre PIC et PSOC

Le PIC ne possède pas de circuits logiques ou analogiques configurables, en conséquence lorsque l'on développe une application, en fonction de ses besoins, on doit choisir son circuit dans toute une série de PICs avec des variantes spécifiques. Avec un PSoC, un seul modèle peut répondre à une multitude d'application.