MICROINFORMATIQUE

Chronomètre

# Introduction

Ce projet a pour objectif de concevoir une horloge/chronomètre et de l’implémenter sur une carte de laboratoire. Le système demande l’utilisation des principales fonctionnalités d’un microcontrôleur. Les projets nécessitent la mise en œuvre d’une approche de conception structurée, le code devra être modulaire.

# Mode d’emploi

Les boutons 1,2,3 et 4 correspondent respectivement à SW1, SW2, SW3, SW4

* Le bouton 1 change le mode d’affichage entre l’horloge et le chronomètre.
* Le bouton 2 arrête/démarre le chronomètre mais seulement lorsque l’affichage est en mode chrono.
* Le bouton 3 remet à zéro le chronomètre mais seulement lorsque l’affichage est en mode chrono.
* Le bouton 4 lance la synchronisation de l’horloge par USB, en vérifiant d’abord que la connexion est établie, faute de quoi un message d’erreur est affiché. Il faut ensuite écrire l’heure courante en format HH :MM et par la suite valider la synchronisation en envoyant le caractère I.

# Description générale

Le projet possède les fonctionnalités suivantes :

* Gestion de sept variables temporelles, pour l’horloge (mode CK Clock) et le chronomètre (SW).
* Résolution temporelle de 1/100 de seconde.
* Mise à jour et maintien de l’information horaire.
* Affichage de l’heure (CK) ou du chronomètre (SW) sur demande de l’utilisateur.
* Mise à zéro du chronomètre(SW)
* Comptage en mode chronomètre, sur les variables correspondantes.
* Modification de l’heure de puis le PC.
* Gestion des fonctions au moyen des boutons.

# Information sur chaque partie de code

Chaque sous-programme contient ses propres variables. Ce programme ne contient pas de variables globales ce qui permet une modularité complète des éléments. Le programme complet se trouve en annexe.

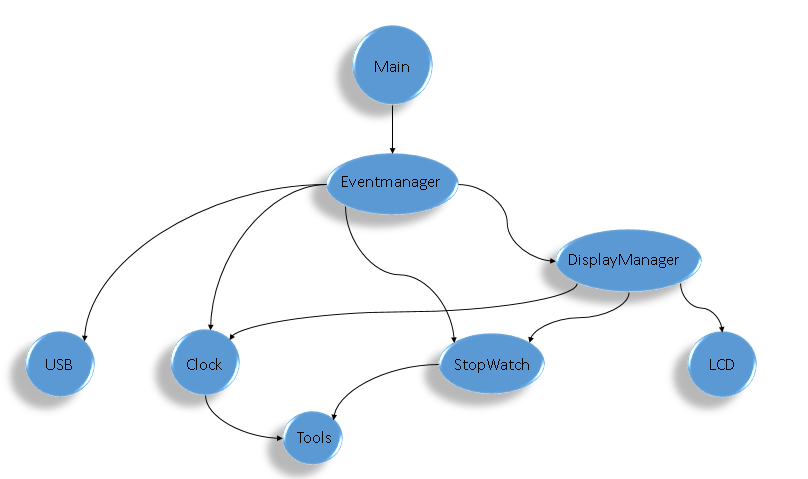


Figure : Schéma de dépendance entre les modules

## Main

* Initialise les paramètres des différents ports, timers, interruptions et appel.
* Initialise les différents modules.
* Appel les fonctions de event manager concernant la gestion du pas temporelle et des boutons ( em\_tick, em\_onPress)

## Event manager

L’event manager gère comme son nom l’indique les évènements, ici les interruptions. Il y a 2 types :

* Les interruptions du TimerA utilisé pour définir la base temporelle
* Les interruptions des boutons qui sont ensuite classées et traitées à l’aide d’une machine à état fini.

C’est le module le plus important, car il contient l’état général de tout le système. Il gère donc les appels aux différents sous-modules comme stopwatch, clock, display manager.

De plus des informations sur la connectivité USB sont affichées pour confirmer à l’utilisateur que le MSP430 est bien connecté. Il indique aussi le débranchement de la carte avec le PC.

## Clock

Il contient les valeurs de l’horloge et les incrémente à chaque appel de la fonction tick(). Des LEDs sont allumées à chaque incrémentation afin de vérifier son bon fonctionnement. L’utilisateur a la possibilité de modifier l’heure grâce aux méthodes synchronize() et setTime() qui vérifient que les valeurs entrées sont valides syntaxiquement et rentre dans la plage possible.

## Stopwatch

Module chronomètre. Il mesure le temps écoulé et l’incrémente à chaque appel de la fonction tick(). Des fonctions de remise zéro et stop/start sont disponibles. Des LEDs permettent aussi la vérification du fonctionnement.

## Tools

Outil de conversion du temps depuis des valeurs entières, en chaine ascii affichable sur le LCD.

## Displaymanager

Ce module gère l’affichage. Pour cela, il utilise le LCD et les LEDs. Il permet aussi de montrer des messages à l’utilisateur. Entre autre, il est optimisé pour ne pas rafraîchir l’affichage plus qu’il ne le faut par seconde.

## LCD

Modification à la fin du document afin de pouvoir ajouter des points proprement sur les segments voulu qui contiennent déjà un caractère. (LCD\_dot)

## Anti-rebond

L’event manager gère lui-même l’anti-rebond des boutons à l’aide de temporisation avec les tick du timer.

Annexe (code complet)