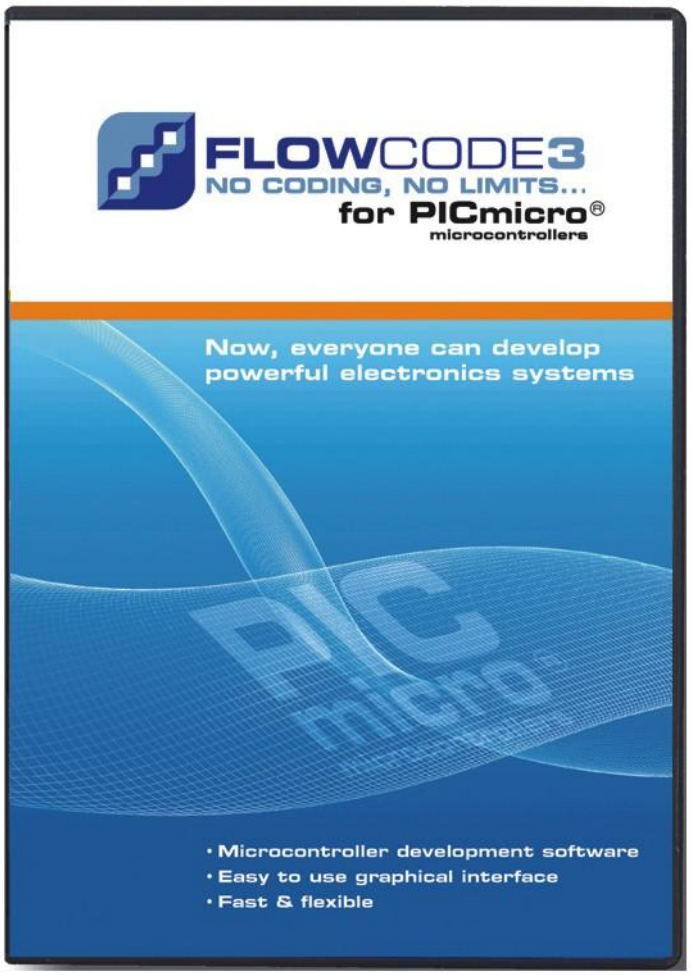
Didacticiel Flowcode pour PIC



16 septembre 2010

V1.2

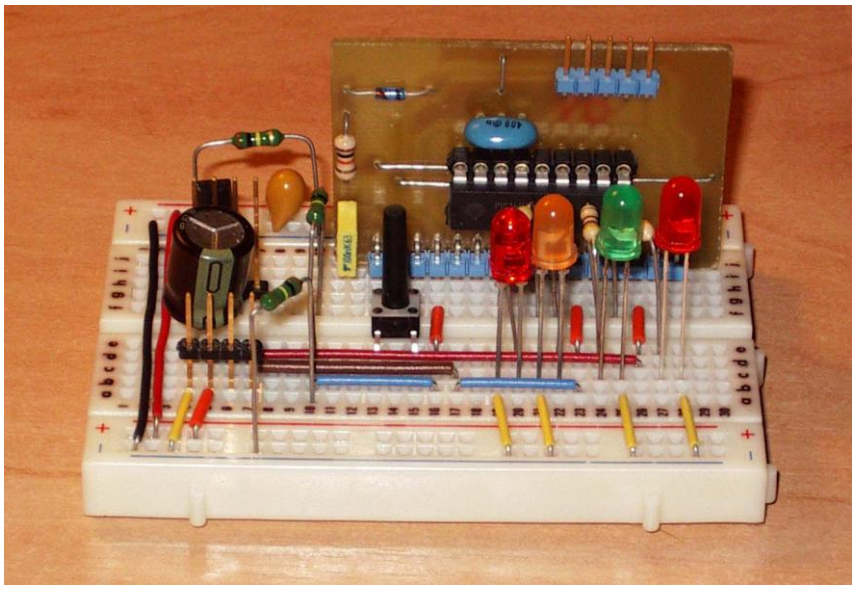
Frédéric GIAMARCHI

frederic.giamarchi@iut-nimes.fr

Département G.E.I.I.

I.U.T. de Nîmes – Université Montpellier II

Sommaire



Projet 1 : Testeur de Batterie 12V

PRESENTATION DE FLOWCODE

Flowcode est un système de programmation de langage haut niveau pour les microcontrôleurs de type PIC ou autres, basé sur l'emploi d'organigramme. Flowcode permet de dessiner et de simuler des systèmes de contrôle ou des contrôles de robots plus ou moins complexes, simplement en dessinant l'organigramme du programme recherché en quelques minutes et sans compétence préalable en programmation.

Flowcode est un langage puissant qui utilise des macros pour faciliter le contrôle de composants complexes comme des afficheurs 7 segments ou LCD, des contrôleurs de moteurs. L'utilisation de macros vous permet de contrôler des composants électroniques complexes sans avoir besoin d'analyser en profondeur la programmation associée.

Flowcode est un produit de la société Matrix [1] et commercialisé par Multipower [2].

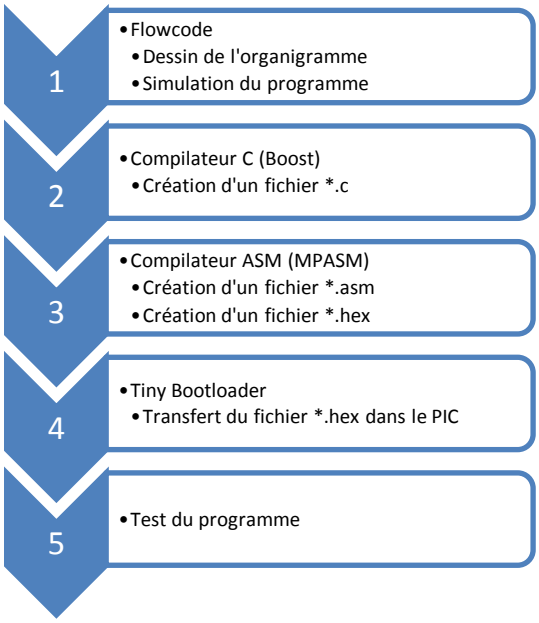
CARACTERISTIQUES

- Ne nécessite pas d'expérience en programmation
- Permet de réaliser rapidement des programmes complexes pour microcontrôleurs
- Utilise les symboles normalisés d'organigramme (ISO-5807)
- Simulation visuelle permettant de debugger pendant le développement
- Produit le code assembleur ASM pour les microcontrôleurs
- Permet d'enseigner la programmation des API
- Permet d'utiliser du code C ou assembleur dans une macro
- Supporte les interruptions et les convertisseurs A/D

PRINCIPE DE FONCTIONNEMENT DE FLOWCODE

Flowcode pour PIC est construit autour d'un compilateur C (Boost-C). Il s'agit d'un compilateur 8/16 bits d'usage général spécialisé pour les microcontrôleurs de type PIC.

Flowcode génère un fichier de code en C à partir de l'organigramme que vous dessinez. Ce code est automatiquement compilé en code assembleur par le compilateur Boost-C et transformé en code machine de type hex avec l'assembleur MPASM de Microchip.



PROGRAMMATION D'UN MICROCONTROLEUR PIC

Le code de type *.hex obtenu par le compilateur de Flowcode peut-être utilisé pour programmer un microcontrôleur de type PIC.

Dans les exemples, nous choisirons le modèle 16F88 que vous aurez la possibilité de programmer lors des projets proposés en fin de document.

Pour cela, le logiciel Flowcode doit être paramétré afin de lancer un utilitaire de programmation. Nous utilisons le logiciel TinyBootloader [3] qui programme un µC PIC par une liaison série RS232.

Pour les nouveaux PC qui ne sont plus équipés d'une liaison série, nous utilisons un câble USB avec convertisseur intégré capable de créer une liaison série virtuelle.

IMPLANTATION DU PROGRAMME DANS LE COMPOSANT

✓ Après avoir simulé, sauvegardé et compilé votre programme, vous pourrez le transférer dans la puce. La vitesse d'horloge doit être de 8000000 Hz (8MHz).

✓ Cliquez sur l'icône  pour lancer la programmation du composant. Appuyez et maintenez enfoncé le bouton de raz de la carte cible.

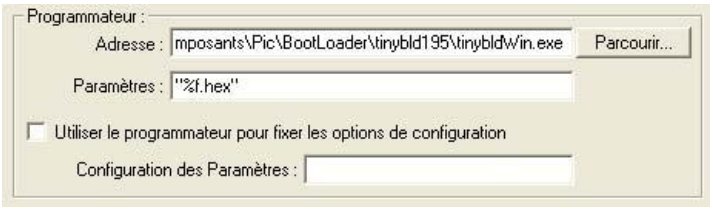
✓ Lorsque le logiciel Tiny Bootloader lance la recherche du composant, relâchez le bouton raz de la cible.

✓ Observez le bon fonctionnement de votre programme.

✓ En cas de non fonctionnement, vérifier le port Comm et le débit (19200 bauds).

NOTES :

Flowcode doit être paramétré pour lancer l'utilitaire de programmation Tiny Bootloader. Dans ***Puce***, sélectionnez ***Options de compilation…*** Remplir les champs suivants :



Le PIC doit, au préalable, avoir été programmé avec un petit bout de programme que l'on appelle un bootloader. Cette opération n'est pas détaillée ici. Pour de plus amples renseignements, contacter l'auteur de ce document [4].

PREMIER PROGRAMME

OBJECTIFS

Apprendre à utiliser l'environnement ***Flowcode***

Apprendre à lier le programme au composant

DOCUMENTS

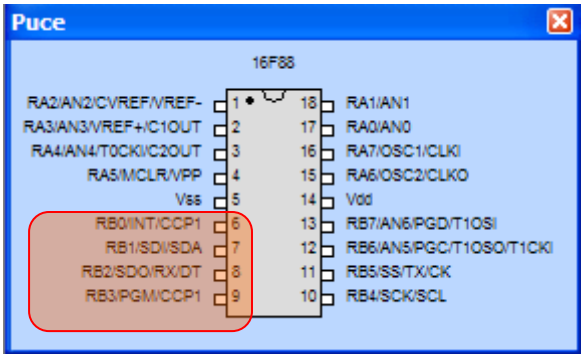
Data sheet 16F88

EXERCICE 1

- Dans ce premier exercice, nous allons créer un programme qui fait clignoter 4 dels connectées au port B d'un microcontrôleur PIC 16F88.

✓ Lancer le programme Flowcode V3

✓ Créer un nouveau document. Choisir le composant à programmer (Ici il s'agit d'un 16F88)



✓ Nous allons relier 4 Dels aux lignes 0 à 3 Port B du composant 16F88. Pour cela, cliquer sur l'icône LEDs .

✓ Cliquer sur la flèche et sur ***Connexions***. Choisir le Port B et Terminer. Cliquer à nouveau sur la flèche et sur ***Propriétés***. Choisir 4 Dels bleues, orientées verticalement.

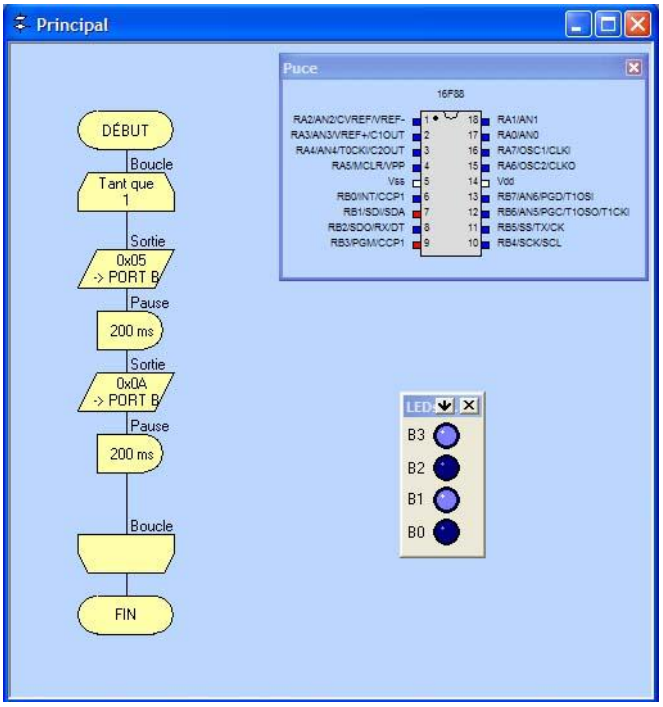
✓ Sélectionner et placer : ***Insérer une boucle***, puis ***Insérer une sortie***, puis ***Insérer une pause*** .

✓ Paramétrer la sortie en double cliquant sur le symbole de sortie et sélectionner le Port B. Paramétrer la pause pour un délai de 200ms.

✓ Sélectionner les deux symboles ensemble, copier les et coller les dessous. Changer les valeurs pour chaque sortie : 0x05 pour la première sortie et 0x0A pour la deuxième.

✓ Sauvegarder le programme sous Exo_1.fcf.

Vous devez obtenir l'organigramme suivant et les éléments associés :



SIMULER LE PROGRAMME :

- ✓ Cliquer sur l'icône ***Exécuter*** . Constater le bon fonctionnement du programme (clignotement des Dels, mais pas des broches du composant), puis arrêter le programme avec l'icône ***Arrêter la simulation*** .
- ✓ Pour visualiser les changements sur les broches du composant, il faut ralentir la vitesse du simulateur. Pour cela, cliquer sur ***Puce***, puis ***Vitesse d'horloge***. Changer la ***Vitesse de Simulation*** pour 2 Hz.
- ✓ Observer le résultat. Essayer avec d'autres vitesses.
- ✓ On utilisera cette astuce pour ralentir le déroulement d'un programme afin de détecter et corriger les erreurs de programmation.

EXERCICES COMPLEMENTAIRES

- ➢ Exo_1a : Modifier le programme pour obtenir un chenillard sur les 4 Dels du port B (aller retour).
- ➢ Exo_1b : Modifier le programme pour compter de 0 à 6 sur un afficheur LED 7 segments (Led7Seg) . L'anode sera reliée à la broche 0 du port A et mise à 1, avant la boucle.

NOTES :

Utiliser Internet pour connaître les combinaisons des segments à éclairer afin d'afficher des chiffres. Remplacer les valeurs en hexadécimal (0xAA) par les valeurs en binaire (0b10101010).

UTILISATION DES VARIABLES

OBJECTIFS

Apprendre à manipuler les variables

Apprendre à lire les entrées du microcontrôleur.

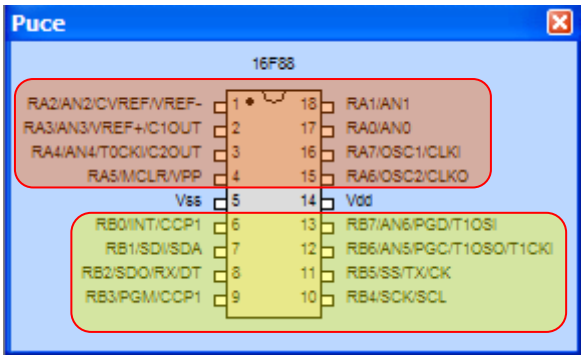
DOCUMENTS

Data sheet 16F88

EXERCICE 2

- Dans ce nouvel exercice, nous allons mémoriser l'état des entrées dans une variable et à afficher le contenu en sortie.
- Nous utiliserons des boutons poussoirs comme entrées sur le port A et des Dels comme sorties sur le port B.

✓ Créer un nouveau document. Choisir le composant à programmer (Ici il s'agit d'un 16F88)



✓ Nous allons relier 8 Dels jaunes au Port B et 8 interrupteurs au Port A. Pour cela, utiliser les icônes LEDs et Switchs (interrupteurs) . Pour les Switchs, sélectionner le type bascule dans les propriétés.

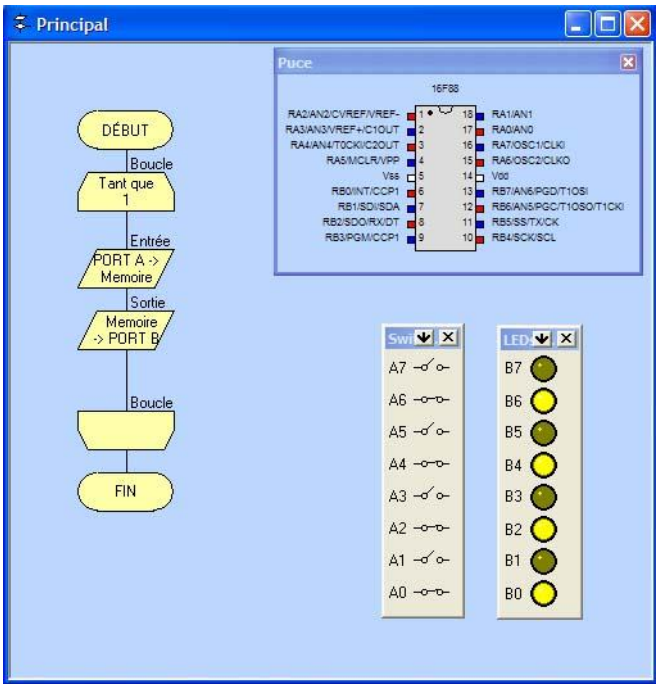
✓ Sélectionner et placer : ***Insérer une boucle***, puis ***Insérer une entrée***, puis ***Insérer une sortie***.

✓ Paramétrer les entrées en double cliquant sur le symbole d'entrées et sélectionner le Port A. Ensuite cliquer sur *Variable* et sur ***Ajouter Variable*** pour créer une variable de type Octet. Donner un nom à cette variable, ici Memoire. Ensuite, cliquer sur ***Utiliser Variable***.

✓ Paramétrer les sorties en double cliquant sur le symbole des sorties et sélectionner le Port B. Ensuite sélectionner dans la liste déroulante Variable/valeur une des variables existantes, ici Memoire.

✓ Sauvegarder le programme sous Exo_2.fcf.

Vous devez obtenir l'organigramme suivant et les éléments associés :



SIMULER LE PROGRAMME :

✓ Cliquer sur l'icône ***Exécuter***. Constater le bon fonctionnement du programme en manipulant les divers interrupteurs A0 à A7 et en regardant le contenu de la variable Memoire.

EXERCICES COMPLEMENTAIRES

- Exo_2a : Modifier le programme pour obtenir l'inverse, Led allumée pour interrupteur ouvert. Ajouter un symbole ***Calcul*** entre les symboles Entrée et Sortie, puis manipuler la variable Mémoire pour obtenir le résultat souhaité.
- Exo_2b : Ajouter un afficheur LED 7 segments (Led7Seg) à coté des Dels. Manipuler les interrupteurs pour faire afficher les chiffres.
- Exo_2c : Modifier l'exercice précédent en utilisant une variable tableau pour enregistrer les combinaisons des chiffres. Vous pouvez afficher en hexadécimal.

NOTES :

On peut créer un tableau de variables en écrivant : Table[10] pour un tableau à 10 chiffres.

En commençant par 0, on peut affecter une valeur : Table[0] = 0b10001010

STRUCTURE DECISIONNELLE

OBJECTIFS

Apprendre à utiliser une structure décisionnelle de type : ***si, alors, sinon***

DOCUMENTS

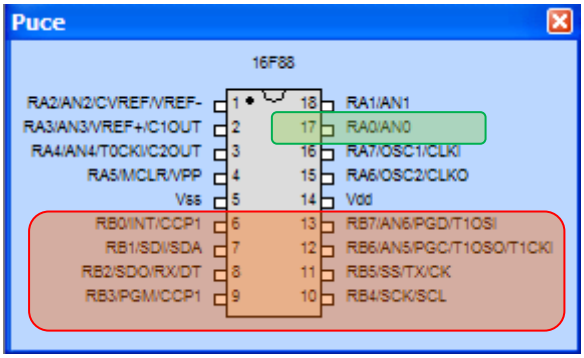
Data sheet 16F88

Liste des opérateurs

EXERCICE 3

Dans ce nouvel exercice, nous allons créer un programme de type chenillard sur 8 Dels connectées au port B du microcontrôleur. Mais le sens de défilement du chenillard sera défini par l'état d'un poussoir connecté sur la broche 0 du port A.

✓ Créer un nouveau document.



✓ Nous allons relier 8 Dels jaunes au Port B et 1 poussoir au Port A. Pour cela, utiliser les icônes LEDs et Switchs (interrupteurs) . Pour les Switchs, sélectionner le type poussoir dans les propriétés.

✓ Sélectionner et placer : ***une boucle***, puis ***une entrée***, puis ***une décision*** puis ***une sortie*** et pour finir ***trois calculs***.

✓ Paramétrer l'entrée en double cliquant sur le symbole d'entrée et sélectionner le Port A. Ensuite cliquer sur *Variable* et sur ***Ajouter Variable*** pour créer une variable de type Octet. Donner un nom à cette variable, ici : **Etat**. Ensuite, cliquer sur ***Utiliser Variable***.

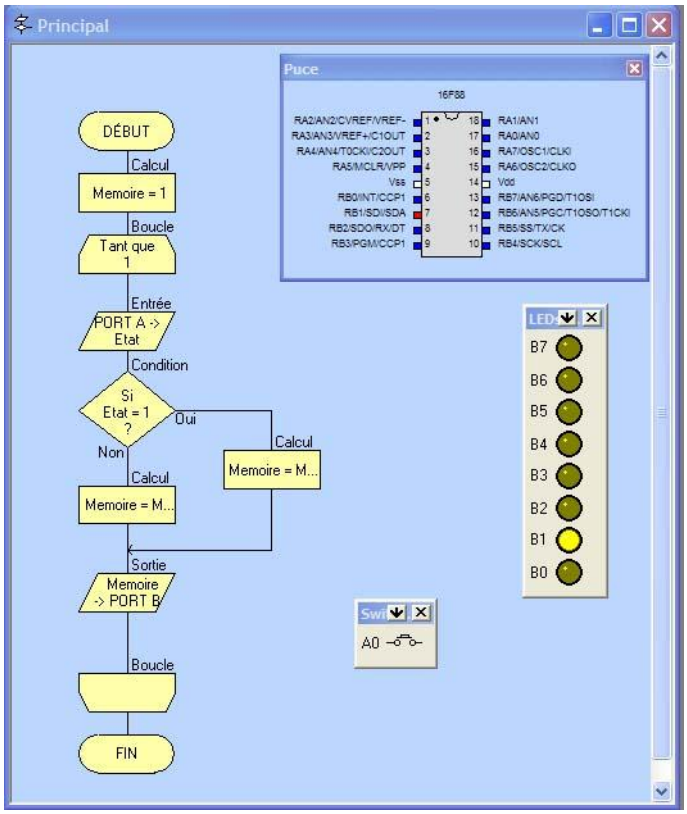
✓ Paramétrer la décision en double cliquant sur le symbole Décision, puis sur *Variable* et sélectionner la variable Etat, puis cliquer sur ***UtiliserVariable***. Compléter avec pour obtenir la condition : **Etat = 1**.

✓ Paramétrer les sorties en double cliquant sur le symbole des sorties et sélectionner le Port B. Ensuite cliquer sur *Variable* et sur ***Ajouter Variable*** pour créer une nouvelle variable de type Octet. Donner un nom à cette variable, ici : **Memoire**. Ensuite, cliquer sur ***Utiliser Variable***.

- ✓ Les calculs vont modifier le contenu de la variable Mémoire. Dans le premier, avant la boucle, on écrira : Memoire = 1. Dans le deuxième, sur une des deux branches du symbole Décision, on écrira : Memoire = Memoire >> 1. Et dans le dernier, sur l'autre branche du symbole Décision, on écrira Memoire = Memoire << 1

- ✓ Sauvegarder le programme sous Exo_3.fcf.

  - Vous devez obtenir l'organigramme suivant et les éléments associés :



SIMULER LE PROGRAMME :

- ✓ Cliquer sur l'icône ***Exécuter***. Constater le bon fonctionnement du programme.

- ✓ Appuyer sur le poussoir A0 pour changer le sens du défilement.

- ✓ Vous pouvez constater un défaut évident du programme. Le défilement disparaît lorsque la Del extrême s'éteint.

EXERCICES COMPLEMENTAIRES

- ➢ Exo_3a : Compléter le programme pour corriger le défaut indiqué précédemment. Utiliser d'autres décisions pour détecter les valeurs extrêmes.

NOTES :

Vous pouvez associer une touche du clavier numérique de l'ordinateur avec le poussoir A0. Aller dans le paramétrage du poussoir A0, dans Connexions et Touche clavier, puis sélectionner Switches(0).

MACRO D'UN COMPOSANT

OBJECTIFS

Apprendre à utiliser les routines composants. Il s'agit de macros associées à des composants permettant de les configurer et de les utiliser de façon simplifiée.

Apprendre à lire des entrées analogiques.

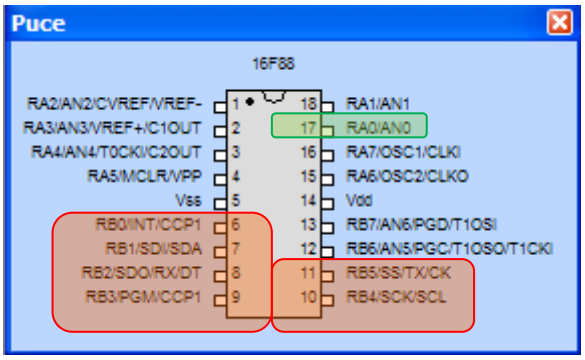
DOCUMENTS

Data sheet 16F88

EXERCICE 4

- Dans ce nouvel exercice, nous allons réaliser un thermomètre à affichage LCD.

✓ Créer un nouveau document.



✓ Nous allons relier l'afficheur LCD aux lignes de 0 à 5 du Port B et le thermomètre sur la ligne 0 du Port A. Pour cela, utiliser les icônes LCDDisplay et Thermometer.

✓ Sélectionner et placer : ***trois routines composants***, ***une boucle***, puis ***quatre routines composants*** dans la boucle**.**

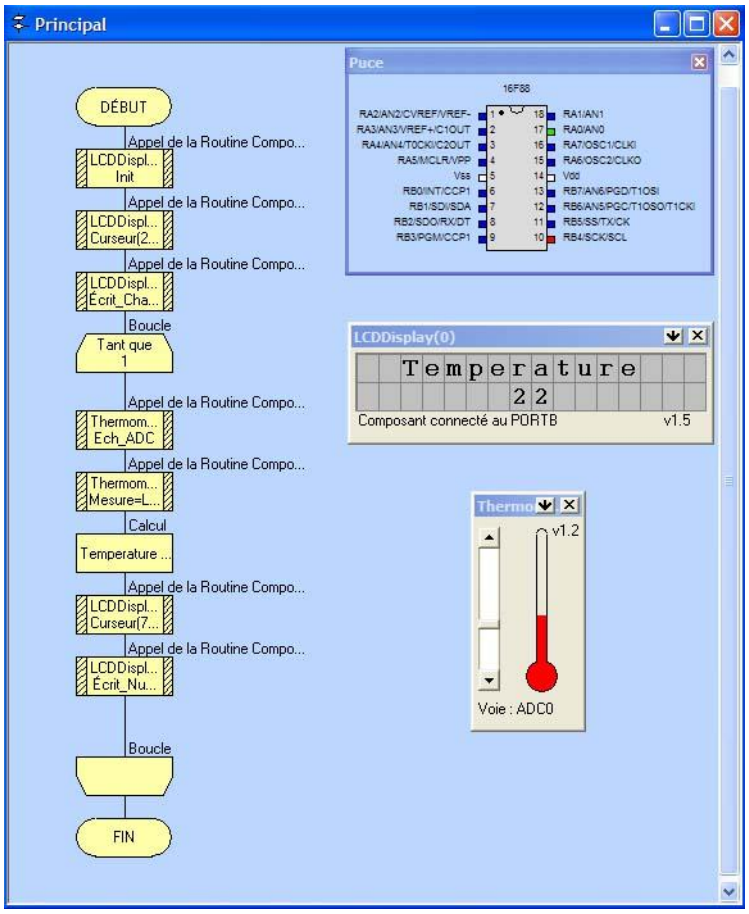
✓ Paramétrer les trois premières routines composants pour l'afficheur LCD afin d'initialiser, de sélectionner la position 3 sur la 1ère ligne et d'afficher un texte.

✓ Paramétrer les deux premières routines composants dans la boucle pour le thermomètre, afin de déclencher une mesure et d'enregistrer cette valeur dans une variable : **Mesure**.

✓ Paramétrer les deux dernières routines composants pour l'afficheur LCD afin de sélectionner la position 7 sur la 2ème ligne et d'afficher la variable Mesure.

✓ Sauvegarder le programme sous Exo_4.fcf.

Vous devez obtenir l'organigramme suivant et les éléments associés :



SIMULER LE PROGRAMME :

- ✓ Cliquer sur l'icône ***Exécuter***. Constater le bon fonctionnement du programme.
- ✓ Identifier le problème sur l'affichage.

EXERCICES COMPLEMENTAIRES

- ➢ Exo_4a : Modifier le programme pour mesure des températures de – 50 à +100°C. Faire apparaître le °C à la suite du chiffre.
- ➢ Exo_4b : Modifier le programme pour utiliser une macro afin regrouper l'initialisation de l'afficheur LCD et une autre pour l'affichage du résultat.

NOTES :

Les routines composants rassemblent des lignes de code réalisant une des fonctions du composant.

MACRO UTILISATEUR

OBJECTIFS

Apprendre à utiliser les macros en général. Il s'agit de créer des sous-programmes pour améliorer la lisibilité du programme principal.

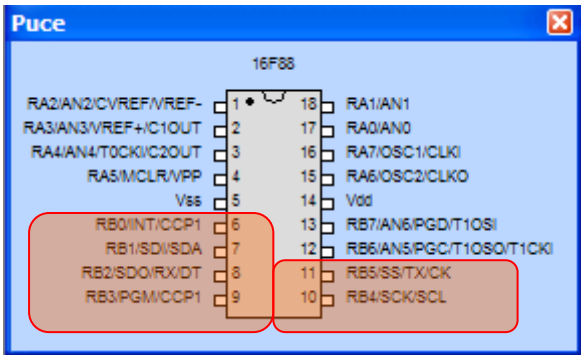
DOCUMENTS

Data sheet 16F88

EXERCICE 5

- Dans ce nouvel exercice, nous allons découvrir les macros utilisateurs et les sous programmes d'interruption.

✓ Créer un nouveau document.



✓ Nous allons relier l'afficheur LCD 4 lignes aux lignes de 0 à 5 du Port B.

✓ Sélectionner et placer : ***une macro*** , ***une boucle*** , et dans la boucle ***un calcul*** ***et une macro*** .

✓ Double cliquer sur la 1ère macro puis sur ***Nouvelle Macro***. Donner un nom à la nouvelle macro, ici **Init**. Puis cliquer sur ***Ok & Editer***. Une nouvelle fenêtre apparaît, dans laquelle, vous devez placer quatre macros composants ***LCDDisplay*** et paramétrer pour afficher Test Macros en 1ère ligne et Nb = en 3ème ligne.

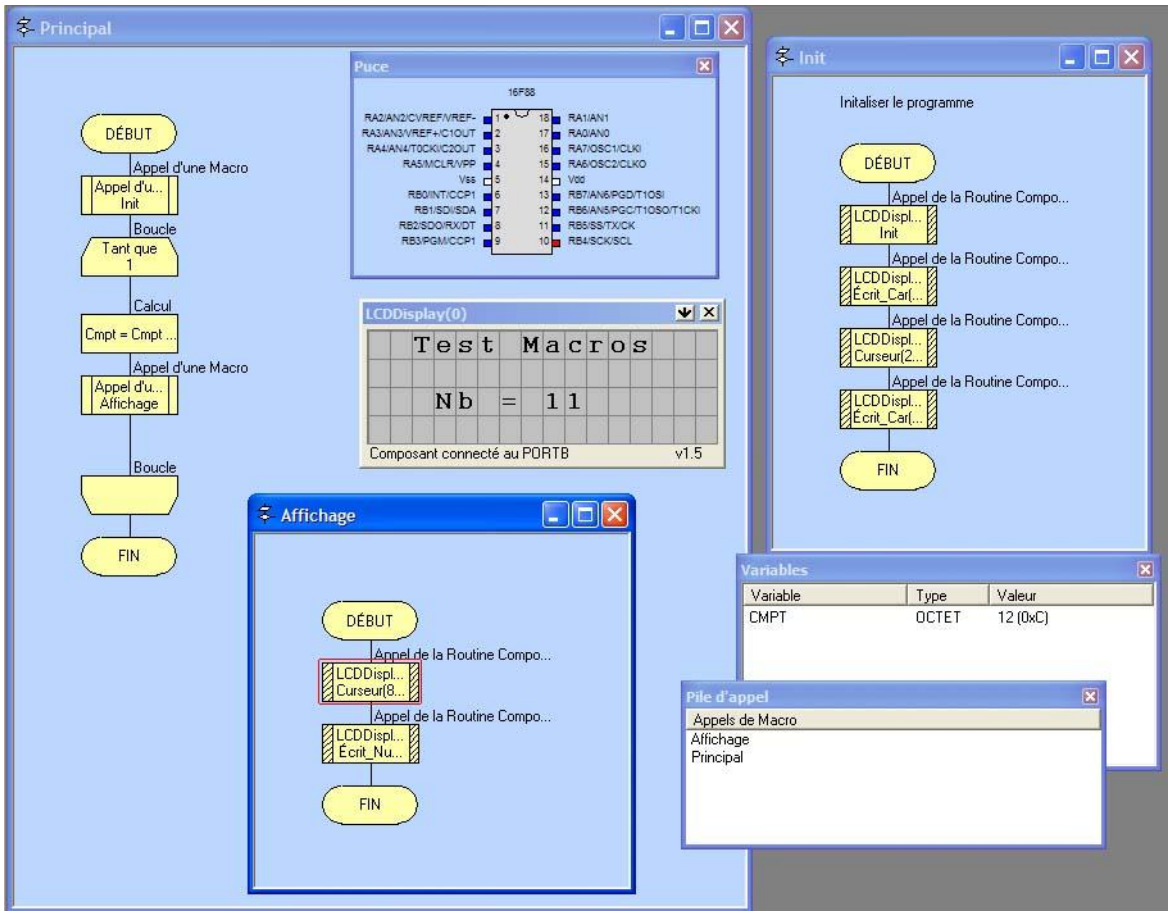
✓ Dans le symbole calcul, créer une variable compteur que vous incrémenter.

✓ Pour la 2ème macro, vous choisirez un nom, Affichage et configurez l'afficheur pour faire apparaître le contenu du compteur après le signe = sur la 3ème ligne.

✓ Vous devez créer une macro avec un paramètre. Cliquez sur ***Editer les paramètres*** et ***Ajouter un paramètre*** Nombre. Ce paramètre va apparaître parmi les variables de la macro composant ***LCDDisplay***.

✓ Sauvegarder le programme sous Exo_5.fcf.

- Vous devez obtenir l'organigramme suivant et les éléments associés :



SIMULER LE PROGRAMME :

- ✓ Cliquer sur l'icône ***Exécuter***. Constater le bon fonctionnement du programme.
- ✓ Observer la pile d'appel des macros

EXERCICES COMPLEMENTAIRES

- ➢ Exo_5a : Modifier le programme pour réaliser un chronomètre en minutes et secondes.

NOTES :

SOUS PROGRAMME D'INTERRUPTION

OBJECTIFS

Apprendre à créer un sous programme d'interruption et à la paramétrer. Un sous programme d'interruption permet d'interrompre le programme principal pour exécuter une routine prioritaire.

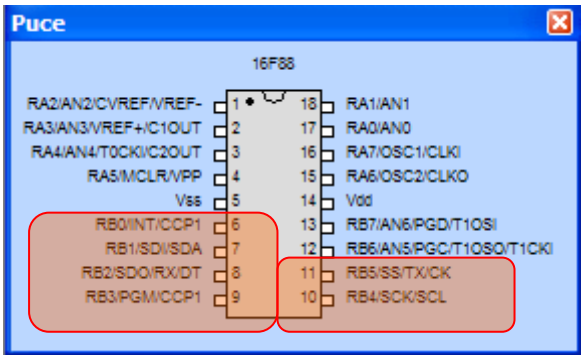
DOCUMENTS

Data sheet 16F88

EXERCICE 6

- Dans ce nouvel exercice, .nous allons réaliser une horloge minutes, secondes, 1/100 de secondes. Il faut configurer une ressource interne du PIC, il s'agit d'un compteur appelé Timer0.

✓ Créer un nouveau document.

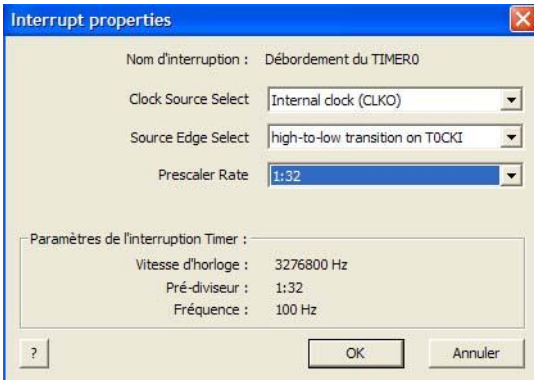


✓ Nous allons relier l'afficheur LCD 2 lignes aux lignes de 0 à 5 du Port B.

✓ Sélectionner et placer : ***une macro*** , ***une interruption*** , ***une boucle*** , et dans la boucle ***une macro*** .

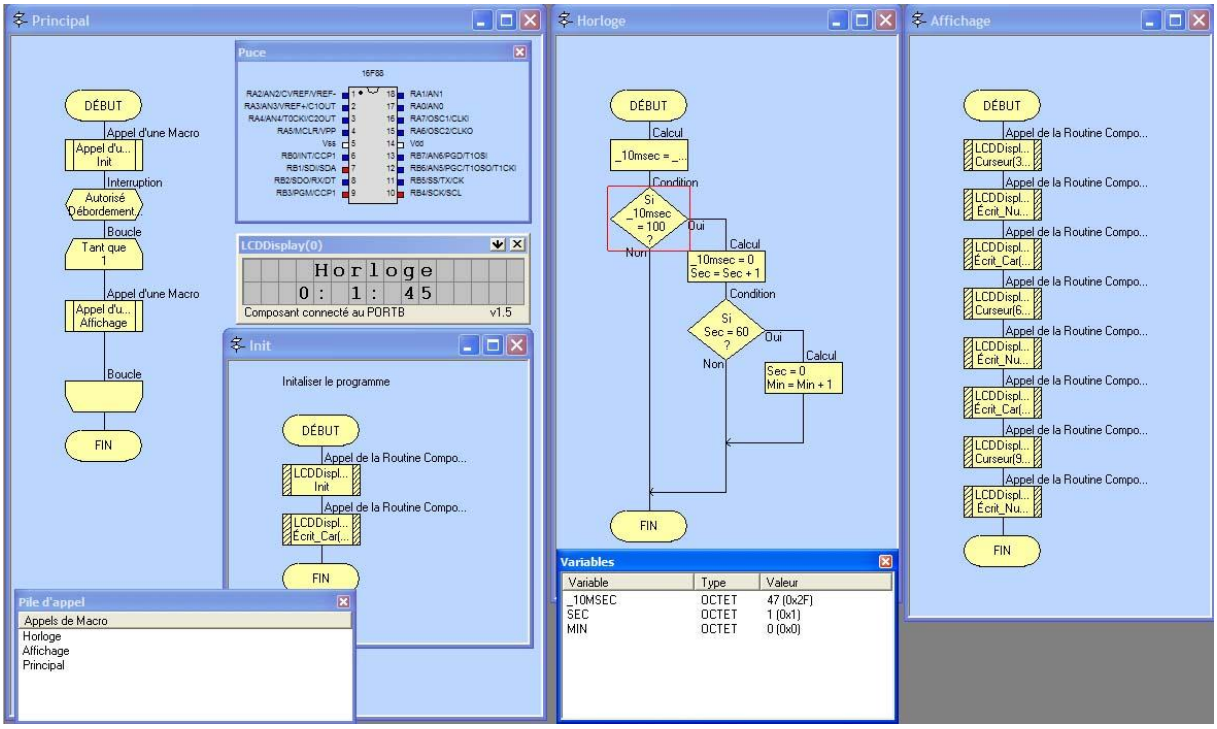
✓ Pour la 1ère ***Macro***, vous initialiserez l'afficheur LCD.

✓ Pour l'interruption, double cliquez dessus et sélectionner comme ***Source***, Débordement du Timer0. Puis sélectionner ***Propriétés*** et compléter comme suit.



- ✓ Dans la nouvelle macro que vous nommerez Horloge, Créer l'organigramme pour compter les 1/100 de secondes puis les secondes jusqu'aux minutes.
- ✓ Dans la 2ème macro, vous afficherez le résultat comme sur l'image suivante.
- ✓ Sauvegarder le programme sous Exo_6.fcf.

Vous devez obtenir l'organigramme suivant et les éléments associés :



SIMULER LE PROGRAMME :

- ✓ Cliquer sur l'icône ***Exécuter***. Constater le bon fonctionnement du programme.
- ✓ Observer la pile d'appel des macros

EXERCICES COMPLEMENTAIRES

- ➢ Exo_6a : Modifier l'horloge de simulation pour avoir la vitesse réelle et ajouter un bouton poussoir pour arrêter le compteur, un autre pour repartir et un autre pour le remettre à zéro.

NOTES :

Modifier l'horloge à 3,2768 MHz afin de générer une interruption toutes les millisecondes.

PROJET : TESTEUR DE BATTERIE

OBJECTIFS

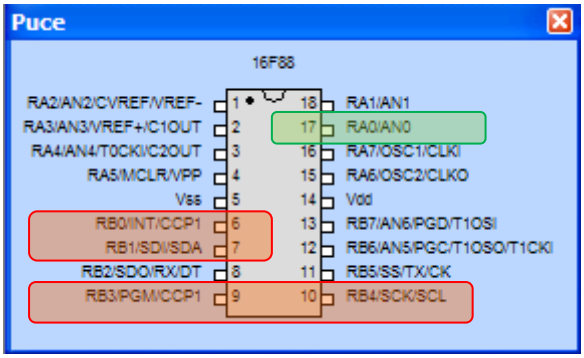
On souhaite indiquer l'état d'une batterie 12V par 4 Dels. Les Dels de couleurs différentes devront clignoter seules pour l'intervalle de tension assigné.

DOCUMENTS

Data sheet 16F88

SCHEMA SIMPLIFIE

- La tension de la batterie est divisée par 3 par un diviseur résistif. La tension réduite est appliquée sur l'entrée 0 du port A. 4 Dels de couleurs Rouge, Jaune, Vert et Rouge sont connectées, respectivement, aux broches 0, 1, 3 et 4 du port B.



✓ Les seuils de basculement d'une Del à l'autre sont définis par le tableau suivant.

✓ Les durées de clignotement des Dels varient en fonction de la couleur de la Del.

Tension	Couleur	Ton	Période
Vbat > 14,4V	Rouge	100ms	300ms
12V < Vbat < 14,4V	Vert	100ms	1s
10,5V < Vbat < 12V	Jaune	100ms	500ms
Vbat < 10,5V	Rouge	100ms	300ms

✓ Sauvegarder le programme sous Projet_1.fcf.

NOTES :

Le programme principal réalisera les tests et vous devez créer une macro pour afficher et faire clignoter les Dels.

PROJET : CAPACIMETRE BATTERIE

OBJECTIFS

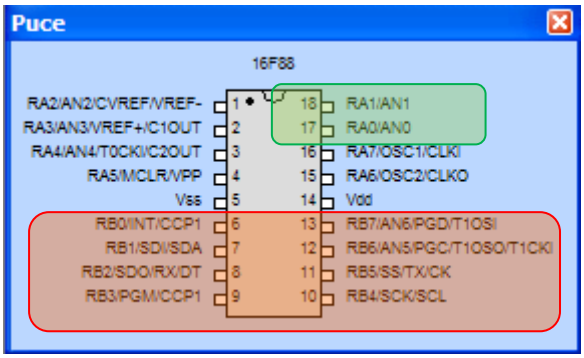
On désire mesurer la capacité d'une batterie 12V. Le résultat devra être affiché sur un afficheur LCD.

DOCUMENTS

Data sheet 16F88

SCHEMA SIMPLIFIE

- La batterie est déchargée dans un puits de courant dont la valeur sera réglée par un bouton ADC, relié à l'entrée 1 du port A. La tension de la batterie est divisée par 3 par un diviseur résistif. La tension réduite est appliquée sur l'entrée 0 du port A. L'afficheur LCD sera connecté au port B.



- ✓ Le calcul et l'affichage sont synchronisés toutes les secondes.
- ✓ Sauvegarder le programme sous Projet_2.fcf.

NOTES :

Modifier l'horloge à 3,2768 MHz afin de générer une interruption toutes les secondes

PROJET : BUGGY

OBJECTIFS

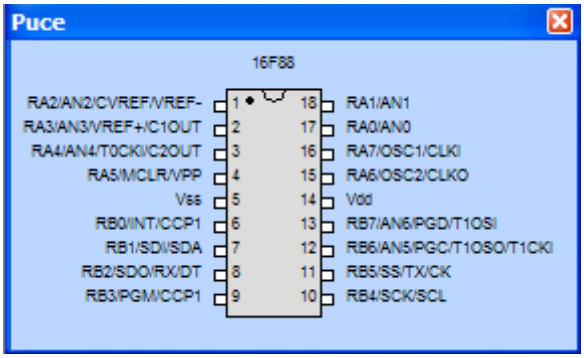
Dans ce projet, nous vous proposons d'écrire l'organigramme qui permettra au Buggy d'atteindre la case d'arrivée. Ce projet, trop simplifié, ne peut pas être testé en réalité sur un vrai robot, mais donne une bonne idée des difficultés de la programmation en robotique.

DOCUMENTS

Data sheet 16F88

SCHEMA SIMPLIFIE

- Le Buggy est relié à diverses lignes par défaut, configurer le au mieux de vos habitudes.



?

✓ Sauvegarder le programme sous Projet_3.fcf.

NOTES :

L'algorithme de base pour la résolution d'un labyrinthe est appelé algorithme de la main droite. Il consiste à suivre le mur de droite. Mais ici, ce n'est pas un vrai labyrinthe.

ANNEXE : OPERATEURS

GENERALITES

Les opérateurs mathématiques, logiques et conditionnels sont utilisés dans les symboles Calcul, Décision.

TABLEAUX DES OPÉRATEURS

Opérateurs mathématiques	Fonction
+ , - , * , / , MOD	Addition, soustraction, multiplication, division, modulo
=	Egal
()	Parenthèses

Opérateurs logiques	Fonction
NOT , AND , OR , XOR	Non , ET , OU , OU exclusif
>> , <<	Décalage à droite, décalage à gauche

Opérateurs décisionnels	Fonction
< , <= , > , >=	Plus petit que, plus petit ou égal à, plus grand que, plus grand ou égal à
= , <>	Egale à , différent de

NOTES :

REFERENCES

[1] Matrix : http://www.matrixmultimedia.com/flowcode.php

[2] Multipower : www.multipower.fr/

[3] Tiny Bootloader ; http://www.etc.ugal.ro/cchiculita/software/picbootloader.htm

[4] Site de l'auteur : www.geii.iut-nimes.fr/fg