Section : ***S***	Option : ***Sciences de l'ingénieur***	Discipline : ***Génie Électrique***

Programmation de la serrure électronique sous Flowcode

Domaine d'application : **Traitement programmé de l'information**	Type de document : **Travaux Pratiques**	Classe : **Terminale**	Date :

I – Problématique à résoudre

La problématique de ce TP consiste à programmer sous Flowcode le microcontrôleur PIC 16f88 de la serrure électronique afin de détecter la saisie du seul code correct autorisant l'ouverture de la porte.

Le clavier (dont seulement 5 touches sont utilisées ici) de la serrure électronique est connecté sur le port A du microcontrôleur :

- La touche RAZ (Remisez A Zéro) sur A0
- La touche 1 sur A1
- La touche 2 sur A2
- La touche 3 sur A3
- Et la touche 4 sur A4

La détection du code correct sera signalée à l'utilisateur par l'allumage d'une LED connectée sur le bit B0 du microcontrôleur.

L'appuie sur la touche RAZ doit annuler à tout moment les chiffres déjà tapés afin de recommencer la saisie du code à partir de zéro (que les chiffres déjà tapés appartiennent au code correct ou pas).

L'appuie sur un mauvais chiffre en cours de saisie aura le même effet que la touche RAZ : retour au début du programme avec annulation de toutes les saisies précédentes et sans en avertir l'utilisateur.

Travail demandé : valider dans le logiciel Flowcode une solution fonctionnelle permettant de détecter la saisie du code correct à 4 chiffres en respectant le cahier des charges suivant :

- Le code est constitué de 4 chiffres pris seulement dans la liste 1, 2, 3 et 4
- Le code peut utiliser plusieurs fois le même chiffre. Exemples de codes valides : **1234**, **4141**, **3332**, **4444**, **2112**, **2413**, etc.
- Le code sera enregistré au début de l'algorigramme dans la variable **code** de type entier (dans un bloc ***Calcul***). Exemple : **code=1234**
- La valeur numérique du port A informant sur les touches enfoncées sera enregistrée dans une variable **clavier** de type octet (dans un bloc ***Entrée*** à insérer dans toutes les boucles d'attente)
- Le tableau suivant rappelle la valeur que prendra la variable **clavier** pour chacune des touches enfoncées. Ces valeurs correspondent naturellement aux puissances de 2 :

Touche enfoncée	**Valeur de la variable `clavier`**
aucune	**0**
RAZ	**1**
1	**2**
2	**4**
3	**8**
4	**16**

Par exemple si le code est **1234** l'algorithme de détection du code correct ressemblera globalement au suivant :

- Attendre que **clavier=2** (touche 1 enfoncée)
- Attendre que **clavier=0** (touche 1 relâchée)
- Attendre que **clavier=4** (touche 2 enfoncée)
- Attendre que **clavier=0** (touche 2 relâchée)
- Attendre que **clavier=8** (touche 3 enfoncée)
- Attendre que **clavier=0** (touche 3 relâchée)
- Attendre que **clavier=16** (touche 4 enfoncée)
- Attendre que **clavier=0** (touche 4 relâchée)
- A tout moment si **clavier=1** (touche RAZ enfoncée) on annule la saisie précédente et on repart à zéro

Mais si le code est modifié et devient **3322** (par exemple) l'algorithme devient alors :

- Attendre que **clavier=8** (touche 3 enfoncée)
- Attendre que **clavier=0** (touche 3 relâchée)
- Attendre que **clavier=8** (touche 3 enfoncée)
- Attendre que **clavier=0** (touche 3 relâchée)
- Attendre que **clavier=4** (touche 2 enfoncée)
- Attendre que **clavier=0** (touche 2 relâchée)
- Attendre que **clavier=4** (touche 2 enfoncée)
- Attendre que **clavier=0** (touche 2 relâchée)
- A tout moment si **clavier=1** (touche RAZ enfoncée) on annule la saisie précédente et on repart à zéro

Le problème suivant se pose alors : comment dans ces conditions faire en sorte que la structure de l'algorigramme ne dépende pas du code ? En d'autre termes, pour changer le code seule l'affectation de la variable **code** dans le premier bloc calcul doit être modifiée (exemple : **code=3242**) **mais la structure de l'algorigramme ne doit en aucun cas être modifiée ou adaptée au nouveau code**.

II – Eléments de solution sous Flowcode

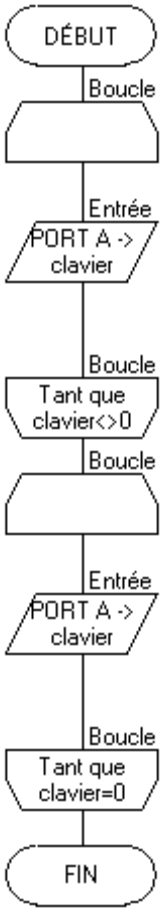
La solution présentée ici est constituée d'un algorigramme principal et de 3 macros. Présentation des 3 macros :

- La macro **attendre** détecte le relâchement de toute les touches suivie d'une touche enfoncée
- La macro **decodage** convertit l'entier code en 4 octets **c1**, **c2**, **c3** et **c4** ayant chacun le poids d'un chiffre
- La macro **puissance** renvoie dans la variable **resultat** la valeur de ***« 2 puissance exposant »***

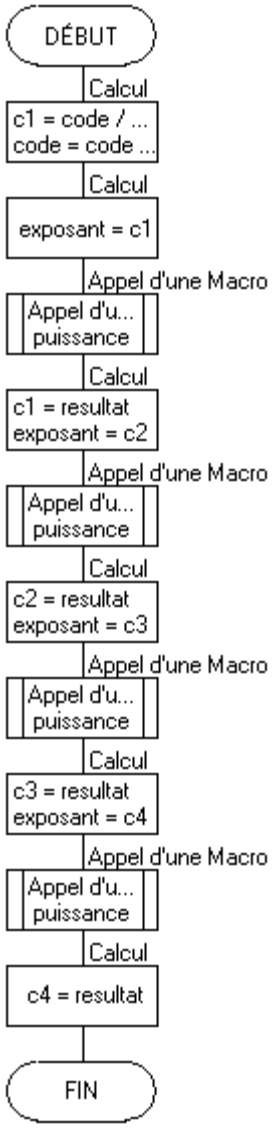
Liste des 9 variables à créer et utilisées dans ce programme :

clavier, c1, c2, c3, c4, n, exposant et **resultat** de type octet

code de type entier



*Algorigramme de la macro **attendre***



*Algorigramme de la macro **decodage***

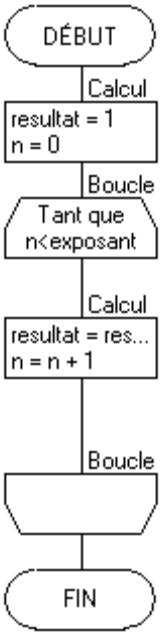
*Contenu complet du premier bloc calcul de la macro **decodage** :*

```
c1 = code / 1000
code = code - 1000 * c1
c2 = code / 100
code = code - 100 * c2
c3 = code / 10
c4 = code - 10 * c3
```

*Contenu complet du second bloc calcul de la macro **puissance** :*

```
resultat = resultat * 2
n = n + 1
```

La macro **puissance** remplace simplement la fonction puissance de 2 qui n'existe pas dans Flowcode.



*Algorigramme de la macro **puissance***

Rôle et explication de la macro `attendre` :

Le rôle de cette macro **`attendre`** est double : attendre que la touche en cours d'utilisation soit relâchée, puis patienter jusqu'à l'appuie de la touche suivante. De plus la macro **`attendre`** actualise en permanence la variable **`clavier`** qui indique **le poids des chiffres** tapés.

Rôle et explication de la macro `decodage` :

Le rôle de cette macro **`decodage`** est de déterminer quel sont les 4 chiffres du code, dans l'ordre. Après analyse de l'entier **`code`**, la macro **`decodage`** enregistre dans les variables **`c1`**, **`c2`**, **`c3`** et **`c4`** **le poids de chacun des chiffres** du code.

- **`c1`** correspond au poids du 1er chiffre
- **`c2`** correspond au poids du 2nd chiffre
- **`c3`** correspond au poids du 3ème chiffre
- **`c4`** correspond au poids du 4ème chiffre

Par exemple si le code est **`1234`**, alors les variables **`c1`**, **`c2`**, **`c3`** et **`c4`** auront pour valeurs :

- **`c1=2`** (car le 1er chiffre est 1)
- **`c2=4`** (car le 2nd chiffre est 2)
- **`c3=8`** (car le 3ème chiffre est 3)
- **`c4=16`** (car le 4ème chiffre est 4)

Si maintenant le code est **`2423`**, alors la macro **`decogade`** affectera les valeurs suivantes aux variables **`c1`**, **`c2`**, **`c3`** et **`c4`** :

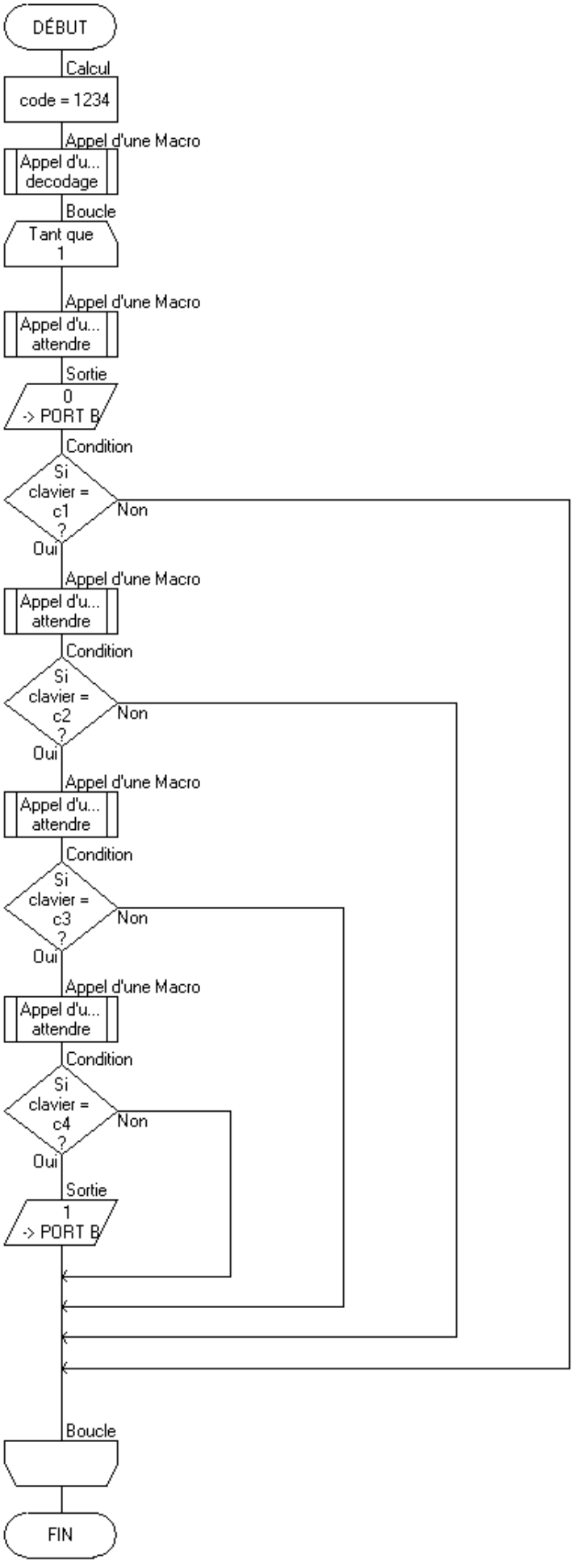
- **`c1=4`** (car le 1er chiffre est 2)
- **`c2=16`** (car le 2nd chiffre est 4)
- **`c3=4`** (car le 3ème chiffre est 2)
- **`c4=8`** (car le 4ème chiffre est 3)

Ainsi le programme principal n'aura plus qu'à attendre que la variable **`clavier`** prenne successivement les valeurs indiquées par les variables **`c1`**, **`c2`**, **`c3`** et **`c4`**.

Commentaires du programme principal :

Après le décodage du code dans les 2 premiers blocs le programme entre dans une boucle infinie dans laquelle :

- la première macro **`attendre`** attend la saisie du premier chiffre
- le premier bloc décision compare ce premier chiffre (renvoyé dans la variable **`clavier`**) avec le premier chiffre du code (enregistré dans la variable **`c1`**)
- si le chiffre tapé est correct on continue, si non on annule tout et on repart à zéro (que la touche enfoncée soit un mauvais chiffre ou la touche RAZ)
- en cas de 1er chiffre correct on attend et on teste le 2nd chiffre (en comparant **`clavier`** à **`c2`**)
- si le 2nd chiffre est faux retour à la case départ, si non on attend et on teste le 3ème chiffre
- si le 3ème chiffre et correct on attend et on teste le 4ème chiffre (en comparant **`clavier`** à **`c4`**)
- si le 4ème chiffre et correct on allume la LED connecté sur le bit 0 du port B
- la LED s'éteindra au prochain appuie sur une touche quelconque parmi les 5



Algorigramme du programme principal appelant les macros