



**HAL**  
open science

## Un projet fédérateur d'Informatique Industrielle en IUT GEII : Programmation d'un GRAFCET en langage C sur PIC 18F452

Didier Crestani, Vincent Creuze, Xavier-François Hochmuth, Eric Maurines,  
Eric Pommier

### ► To cite this version:

Didier Crestani, Vincent Creuze, Xavier-François Hochmuth, Eric Maurines, Eric Pommier. Un projet fédérateur d'Informatique Industrielle en IUT GEII : Programmation d'un GRAFCET en langage C sur PIC 18F452. CETSIS: Colloque sur l'Enseignement des Technologies et des Sciences de l'Information et des Systèmes, Mar 2010, Grenoble, France. lirmm-00463843

**HAL Id: lirmm-00463843**

**<https://hal-lirmm.ccsd.cnrs.fr/lirmm-00463843v1>**

Submitted on 15 Mar 2010

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Un projet fédérateur d'Informatique Industrielle en IUT GEII : Programmation d'un GRAFCET en langage C sur PIC 18F452

Didier CRESTANI, Vincent CREUZE, Xavier-François HOCHMUTH, Eric MAURINES, Eric POMMIER  
Didier.Crestani@univ-montp2.fr ; Vincent.Creuze@univ-montp2.fr ;  
Xavier-Francois.Hochmuth@univ-montp2.fr ; Eric.Maurines@univ-montp2.fr ; Eric.Pommier@univ-montp2.fr  
IUT de Montpellier, Département GEII, 99 Avenue d'Occitanie, 34296 Montpellier

**RESUME :** Afin de préparer les futurs techniciens supérieurs aux réalités technologiques de l'informatique industrielle, le département GEII de l'IUT de Montpellier propose en 1<sup>ère</sup> et 2<sup>nde</sup> année de formation, l'étude et la mise en œuvre d'un support matériel développé autour du PIC18F452. Cet article présente un projet couvrant la plupart des objectifs et notions d'Informatique Industrielle apparaissant dans le Programme Pédagogique National du DUT GEII. Le contenu s'adresse donc à un public en formation BAC+1 et BAC+2 désireux de se familiariser avec l'environnement des microcontrôleurs de la famille PIC18. Le projet est riche en termes de notions abordées : il nécessite donc une présentation progressive et détaillée afin de faciliter l'acquisition de ces notions.

**Mots clés :** maquette pédagogique, microcontrôleur, GRAFCET, langage C

## 1 INTRODUCTION

A la suite de la forte évolution de l'informatique industrielle et de ses applications ces deux dernières décennies, les compétences des techniciens supérieurs ont dû et doivent encore fortement s'enrichir. Les étudiants doivent connaître les matériels disponibles et posséder les savoir-faire permettant de les mettre en œuvre. Parmi l'ensemble des microcontrôleurs utilisés dans l'industrie, le département Génie Electrique et Informatique Industrielle (GEII) de l'IUT de Montpellier [1] a fait le choix de la famille PIC18. Cette cible évoluée présente nombre de fonctionnalités et propriétés permettant de couvrir l'ensemble des notions soulignées dans le Programme Pédagogique National (PPN) du Diplôme Universitaire de Technologie GEII [2]. En accord avec les contenus du PPN, l'équipe pédagogique choisit d'articuler sa formation en 1<sup>ère</sup> et 2<sup>nde</sup> année autour de projets permettant aux étudiants d'une part, de se familiariser avec l'environnement du PIC18 et d'autre part, d'acquérir les principes fondamentaux de la programmation. Cet apprentissage s'effectue grâce à des supports matériels élaborés au département GEII dont le développement prend en compte les nécessaires robustesse, flexibilité et facilité de mise en œuvre qu'ils doivent présenter. Les étudiants ont donc à disposition en séance de TD et TP un outil ergonomique adapté aux objectifs pédagogiques visés.

Cet article est composé de deux parties. La première présente l'environnement pédagogique et technologique dans lequel s'inscrit le projet développé dans cet article. La seconde détaille le cahier des charges et les principales étapes de sa réalisation.

## 2 ENVIRONNEMENT PEDAGOGIQUE ET TECHNOLOGIQUE DU PROJET

### 2.1 Objectifs pédagogiques

Par le biais de l'élaboration d'un projet portant sur la gestion du déplacement d'un chariot mobile, l'équipe pédagogique a eu le souci d'exploiter les fonctionnalités du PIC18 afin d'aborder la plupart des notions et fonctions fondamentales de l'informatique industrielle. Ainsi, l'outil proposé permet aux étudiants non seulement d'appréhender la synthèse de code en langage évolué C ainsi qu'en langage machine, mais aussi d'acquérir la notion d'interruption grâce à l'élaboration de temporisations. Le projet nécessite également la maîtrise de la gestion des entrées/sorties pour communiquer notamment avec les capteurs TOR de la partie opérative. Le projet nécessite aussi la gestion d'une interface homme-machine puisque l'état de la partie opérative doit être visualisé sur un afficheur LCD. Enfin, le cœur de la synthèse porte sur l'élaboration d'un GRAFCET qui fait appel à des fonctionnalités avancées du langage C (structures et unions) implantées dans le PIC18.

De plus, le support matériel utilisé pour le projet permet la mise en œuvre de convertisseurs analogiques-numériques et numériques-analogiques ainsi que l'étude et la gestion d'une communication entre le microcontrôleur et un périphérique obéissant au protocole I2C.

### 2.2 Le microcontrôleur MICROCHIP PIC18F452

Le choix du microcontrôleur a fait l'objet d'une étude menée auprès des industriels locaux. Pour les processeurs 8 bits plus de 80% utilisent du MICROCHIP. Pour mémoire 10 ans auparavant la même proportion utilisait des bases 8051 pédagogiquement plus accessible. Un microcontrôleur plus performant 16 ou 32 bits ne se justifie pas (complexité, coût et mise en œuvre

plus importants). Les critères de choix déterminants dans la gamme sont ensuite la possibilité de mémoire FLASH pour la reprogrammation et une taille de RAM interne suffisante pour avoir la majorité des applications uniquement en structure interne. Les critères de vitesse et de consommation ne sont pas critiques. Le boîtier DIP est un plus (facilité de réalisation de cartes prototypes). Le 18F452 [3] "standard milieu de gamme" est retenu bien qu'il ne dispose pas en interne de convertisseur Numérique/Analogique. Des modèles intégrant du réseau CAN 18F458 ou de l'USB 18F4550 quasi pin-compatibles sont disponibles pour l'évolutivité. Les principales caractéristiques sont :

- Une architecture optimisée pour le langage C,
- Une horloge DC > 40MHz avec PLL,
- Mémoire : 32Ko FLASH, 1,5 Ko RAM
- 4 timers, 2 unités Capture/Compare/PWM,
- Une liaison série synchrone I2C ou SPI,
- Une liaison série asynchrone UART,
- Un convertisseur A/N 10 bits intégré,
- Des ports E/S compatibles avec des LEDs.

L'environnement de développement MPLAB fourni par le constructeur est performant et gratuit. Le compilateur C version étudiant limité, également gratuit, est suffisant pour les applications envisagées. Le coût logiciel est donc entièrement nul. La possibilité de débogage in-situ offerte par la gamme PIC18 a introduit une évolution conséquente dans la "démocratisation" des manipulations : souplesse de débogage sur point d'arrêt directement sur carte cible sans nécessité d'émulateur. Une version simplifiée de programmeur/débogueur compatible avec celui du constructeur est implanté sur la carte qui est ainsi autonome : développement, débogage directement sur cible, flashage permanent d'applications.

### 2.3 Maquette pédagogique

Le KIT\_PIC18\_2007, qui a été conçu et assemblé au sein du département GEII de Montpellier, est principalement composé de deux cartes. La carte mère (fig. 1) est utilisée en TD pour l'apprentissage de la programmation d'un microcontrôleur aussi bien en C qu'en assembleur. La carte fille (fig. 2), en liaison avec la carte mère, permet en TP de relier le kit à diverses maquettes pédagogiques. Le KIT\_PIC18\_2007 est construit autour d'un microcontrôleur PIC18F452 cadencé par un quartz de 10MHz.

La carte mère possède :

- Son propre programmeur/debugger compatible ICD2© permettant une liaison simple (USB) avec un PC,
- Un afficheur LCD 2\*16 caractères (mode 4 bits),
- 8 LEDs de visualisation (Port D du PIC),
- 4 interrupteurs connectés sur 2 entrées d'interruptions externes et sur 2 entrées externes de 2 compteurs,

- 2 entrées analogiques (CAN interne) adaptées pour pouvoir convertir du 0/+5V ou du -5V/+5V (sur fiche BNC),
- 2 sorties analogiques (CNA externe série I2C MAX517) adaptées pour pouvoir convertir en 0/+5V ou en -5V/+5V (sur fiche BNC),
- Un capteur de température intégré DS1621 avec protocole I2C,
- Une mémoire EEPROM externe 24LC256
- Un connecteur BD37 permettant la liaison (25 lignes entrée/sortie sur les 34 du µC) avec la carte fille ou une autre carte de développement (ex : commande d'un robot 5 axes).

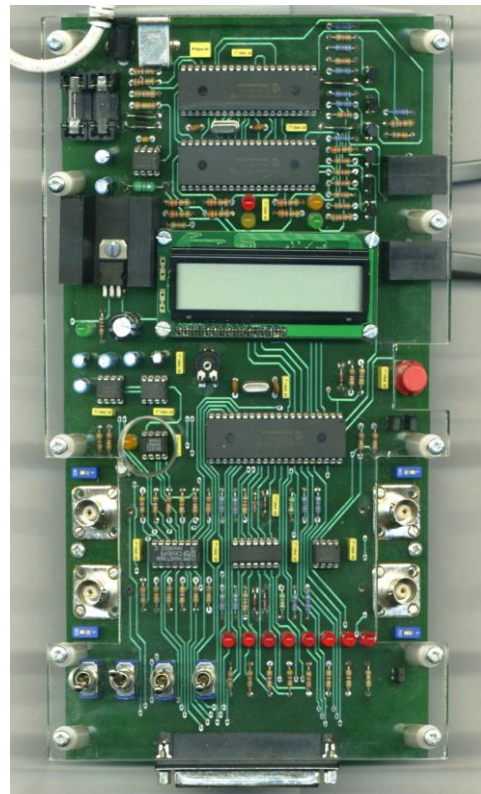


fig 1 : Carte mère du KIT\_PIC18\_2007

La carte fille possède :

- 5 entrées logiques protégées par des buffers (fiche banane 2mm),
- 10 sorties logiques protégées par des buffers (fiche banane 2mm),
- 1 sortie analogique (CNA externe parallèle 8 bits AD7524) adaptée pour pouvoir convertir en 0/+5V ou en -5V/+5V (sur fiche BNC). La liaison parallèle du CNA passant par le port D du PIC permet une lecture du code numérique directement sur les LEDs.

Tous les composants intégrés du kit sont sur supports DIP pour faciliter la maintenance. Le circuit imprimé du kit est inséré entre deux plaques de plexiglas afin de prévenir toutes mauvaises manipulations.

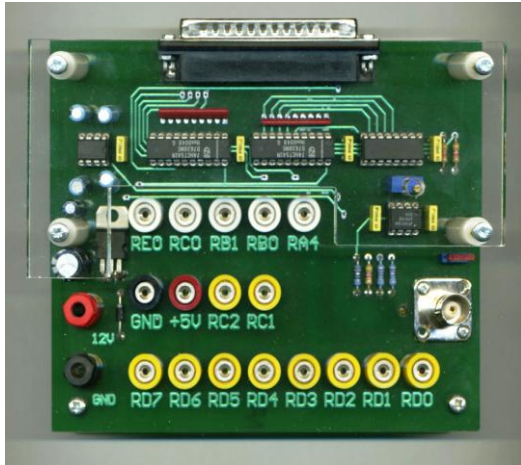


fig 2 : Carte fille du KIT\_PIC18\_2007

Le kit peut fonctionner en mode :

- Debugger, sous MPLAB, en maintenant la liaison entre le programmeur ICD2© et le PIC18F452.
- Programmeur, en supprimant après programmation, la liaison entre l'ICD2© et le PIC18F452.

Les différents thèmes abordés avec le kit sont :

- Commande de Moteur à courant continu (PWM),
- Commande de moteur pas à pas,
- Mesure de la vitesse de rotation d'un moteur (périodemètre ou fréquencemètre),
- Principe de l'oscilloscope numérique (CAN & CNA),
- Lecture d'un badge magnétique,
- Utilisation d'un afficheur LCD 2\*16 caractères alphanumériques,
- Réalisation d'un thermomètre avec affichage de la température (mise en oeuvre du protocole I2C©).
- Programmation en langage C d'un GRAFCET.

### 3 EXEMPLE DE PROGRAMMATION EN LANGAGE C D'UN GRAFCET

Le GRAFCET [4] est un langage normalisé [5] facilitant la spécification et la commande des systèmes automatisés. Les outils mis à disposition actuellement pour programmer les automates font oublier qu'il existe de nombreuses façons de mettre en œuvre un GRAFCET de commande. Le projet que nous allons présenter prend pour prétexte la synthèse de la commande GRAFCET d'un simple chariot sur un microcontrôleur PIC. Il permet de balayer bien des notions abordées par les étudiants. Positionné en fin de première année au début du dernier module d'enseignement "Contrôle Commande des Systèmes Industriels", il est traité sur 3 séances de 3 heures.

### 3.1 Cahier des charges

Le cahier des charges du projet à réaliser s'énonce de la façon suivante :

Un chariot se déplace sur un rail rectiligne (fig. 3). Un capteur Gauche (Droite), actif à l'état haut, détecte sa présence à l'extrémité gauche (droite) du rail. Le chariot est initialement du côté gauche. Un interrupteur Départ\_Cycle lance le cycle de déplacement lorsqu'il prend la valeur 1. Le chariot se déplace alors vers la droite (Ordre Moteur\_Droite). Après avoir atteint cette extrémité, il y demeure 5 secondes avant de revenir à gauche (Ordre Moteur\_Gauche) dans l'attente d'une nouvelle demande de cycle. Les ordres moteur sont des ordres Tout Ou Rien.

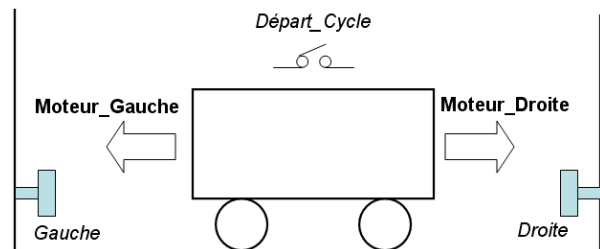


fig 3 : Le chariot à commander

Après avoir établi le GRAFCET correspondant on demande de l'implanter en langage C sur la maquette pédagogique du PIC 18F452. Pour simuler la partie opérative du chariot les conventions suivantes sont adoptées : des interrupteurs reliés aux bits RB0, RB1, RC0 des ports B et C du microcontrôleur sont associés aux capteurs Gauche, Droite et Départ\_Cycle respectivement. Les LEDs 0 et 7 reliées aux bits correspondants du port D permettront de visualiser l'état des ordres Moteur\_Droite et Moteur\_Gauche respectivement. Enfin, pour pallier l'absence d'une visualisation du GRAFCET et d'une partie opérative réelle, l'afficheur LCD sera utilisé pour renseigner sur l'étape GRAFCET active et sur le comportement du chariot.

### 3.2 Principe d'implantation d'un GRAFCET en langage C

Le GRAFCET de commande du chariot est très simple (fig. 4). C'est un cycle linéaire composé de 4 étapes. Une seule des réceptivités fait appel à une relation logique. Les actions correspondent à des ordres continus TOR. On peut simplement noter la présence d'une temporisation lorsque le chariot atteint l'extrémité droite du rail.

La mise en œuvre matérielle ou logicielle d'une machine séquentielle nécessite le respect d'un certain nombre de principes. Le GRAFCET n'y déroge pas. Avant tout, il est indispensable de différencier l'état courant du système de l'état futur qu'il prendra. Dans notre cas, celui-ci permet de rendre compte de l'activité

ou non d'une étape donnée du GRAFCET. Ainsi lors de l'initialisation, il est nécessaire d'une part de définir l'état courant initial (étape(s) initiale(s) active(s)) et d'autre part d'appliquer les commandes correspondant à la position de départ de la partie opérative (Moteur\_Gauche = Moteur\_Droit = 0).

Ensuite, la lecture de l'état des capteurs et leur mémorisation permettent de déterminer quelles conditions d'évolution sont satisfaites (réceptivités vraies) à l'instant de scrutation. L'état futur peut alors être calculé à partir de la connaissance de l'état courant (étapes actives), des conditions d'évolution présentes, de la structure du GRAFCET (lien étape(s) / transition(s) / étapes(s)) en appliquant les règles d'évolution du GRAFCET. Dès lors, il ne reste qu'à exécuter les actions associées aux étapes actives de l'état futur. Avant d'itérer le processus en lisant le nouvel état des capteurs, il convient de désigner l'état futur comme devant l'état courant. On peut évidemment remarquer que toute évolution de la partie opérative entre deux instants de scrutation des capteurs ne peut être pris en compte.

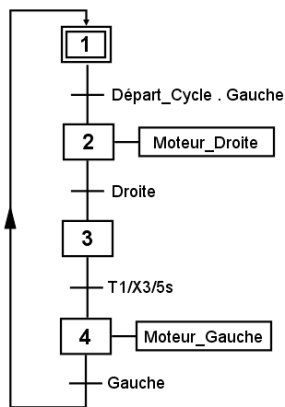


fig 4 : GRAFCET de commande du chariot

Les principes qui viennent d'être présentés sont synthétisés par l'organigramme de la figure 5 qui présente le cycle d'exécution qui doit être suivi lors de l'implantation du GRAFCET sur le microcontrôleur.

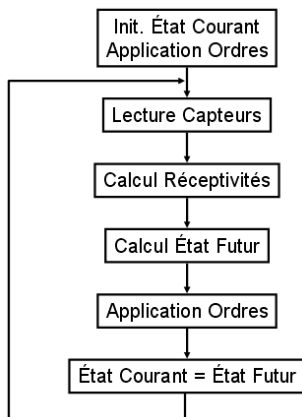


fig 5 : Cycle d'exécution d'une machine séquentielle

### 3.3 Programmation

Quoique simple sur le principe, la mise en œuvre de la boucle décrite ci-dessus implique une structuration rigoureuse du code. Ainsi, nous le décomposons en fonctions, mais surtout nous représentons l'état complet du système en structurant la mémoire comme suit.

#### 3.3.1 Organisation de la mémoire

Pour chacune des variables d'état du système (état d'un interrupteur, état d'une étape, etc.), nous utilisons un bit dans la RAM du PIC. Les bits relatifs à des types de variables d'état identiques sont rassemblés au sein d'octets (états des interrupteurs, états des étapes...).

Sur la figure 6, par exemple, on observe l'octet `Etat_courant` rassemblant les valeurs binaires des quatre étapes du GRAFCET (les quatre autres bits non utilisés sont positionnés à 0). Dans cet exemple, seule l'étape 1 est active (état initial).

Cette structuration facilite la manipulation des variables, conserve la généricité du programme (il suffit de réaffecter les bits pour modifier les capteurs et les actionneurs) et accélère, en fin de cycle, la copie de l'état futur dans l'état présent.

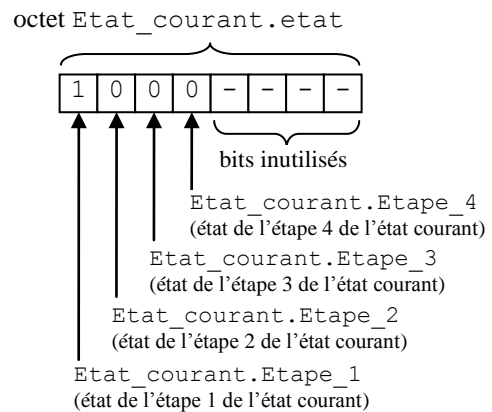


fig 6 : Structuration des variables dans la RAM

#### 3.3.2 Déclaration des variables d'état

Pour exploiter les variables sous la forme décrite précédemment, nous utilisons des « unions », sur le modèle des déclarations des registres du fichier `p18f452.h` de Microchip (PORT, TMR...). Par exemple, l'union `Etat_G7` est utilisée pour les variables `Etat_courant` et `Etat_futur`. Elle est accessible au niveau de l'octet en appelant `Etat_courant.etat` et au niveau du bit en appelant `Etat_courant.EtapeX` (fig. 5).

```
union Etat_G7{
    unsigned char etat;
    struct etat_bf {
        unsigned Etape_1:1;
        unsigned Etape_2:1;
        unsigned Etape_3:1;
        unsigned Etape_4:1;
        unsigned E_vidē:4;
    };
} Etat_courant, Etat_futur ;
```

On procède de la même manière pour les autres variables d'état : entrées, réceptivités, actionneurs. De plus, pour simplifier l'accès des fonctions aux variables d'état, ces dernières sont déclarées en tant que variables globales.

### 3.3.3 Fonctions de gestion du GRAFCET

Les actions de gestion de l'évolution du GRAFCET accomplies à chaque itération de la boucle principale sont décomposées en fonctions déclarées ainsi :

```
void Init_G7(void);
void lecture_entrees(void);
void calcul_receptivites(void);
void calcul_etat_futur(void);
void affectation_sorties(void);
```

Le calcul de l'évolution du GRAFCET est accompli par la fonction `calcul_etat_futur()`. Cette fonction débute par une mise à zéro de l'octet `Etat_futur.etat`, suivie du test de l'état `Etat_present.etat` (actif ou non) de chaque étape (voir exemple ci-dessous). Si une étape est active, la réceptivité associée à la transition "aval" détermine la future étape active.

#### Exemple pour l'étape 1:

```
if(Etat_courant.Etape_1)
{
    if(receptivites.Recept1)
        Etat_futur.Etape_2 = 1;
    else
        Etat_futur.Etape_1 = 1;
}
```

Ce mécanisme permet de traduire à la fois la structuration du GRAFCET de commande et les règles d'évolution du GRAFCET.

### 3.3.4 Fonctions secondaires

Aux fonctions précédentes, s'ajoutent des fonctions secondaires d'initialisation, de gestion de la temporisation et d'affichage.

- *Temporisation*

Pour les temporisations, nous utilisons par défaut les fonctions de la bibliothèque `<delays.h>` incluse dans Microchip C18. Par exemple, `Delay10KTCYx(250)` nous permet d'attendre 1s avec un PIC cadencé à 10MHz.

Aux étudiants souhaitant aller plus loin, nous proposons d'utiliser comme suit le TIMER 1 pour réaliser la temporisation de 5s :

```
#pragma code HIGH_INTERRUPT_VECTOR=0x08
void vecteur(void)
{
    _asm goto tempo_5s _endasm
}
```

```
#pragma code

#pragma interrupt tempo_5s
void tempo_5s(void)
{
    temps_mesure--;
    TMR1H = 0x0B;
    TMR1L = 0xDC;
    PIR1bits.TMR1IF = 0;
    if(temps_mesure == 0)
    {
        TOCONbits.TMR0ON = 0;
        INTCONbits.GIE = 1;
        receptivites.Tempo=1;
    }
}
```

- *Affichage LCD*

Afin de visualiser l'évolution du GRAFCET (étape active), les étudiants utilisent dans un premier temps les LEDs de la maquette. Lorsque le programme fonctionne, nous leur proposons d'utiliser un afficheur LCD déjà étudié durant les travaux d'Etudes & Réalisations (4 bits de données, 3 bits de contrôle, 2 lignes de 16 caractères). Cet afficheur permet d'annoncer le numéro de l'étape active et l'action correspondante, puis de visualiser la position du chariot et son déplacement si nécessaire (voir figure 7). Pour ce dernier point, il faut créer un caractère spécial en forme de chariot lors de l'initialisation du module au début du programme.

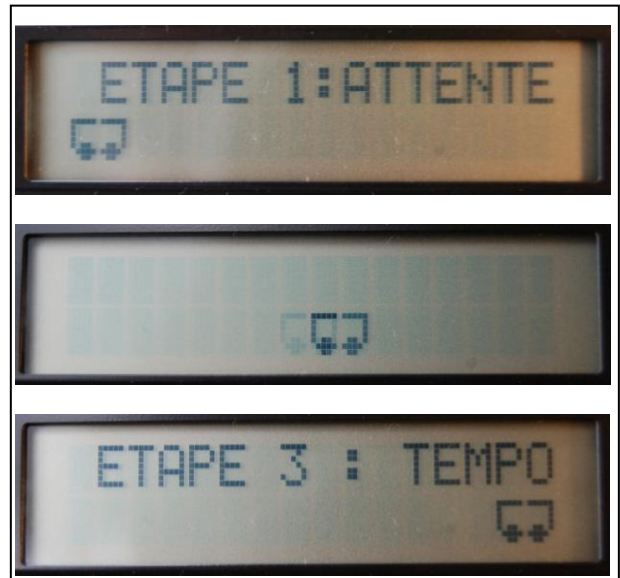


fig 7 : Exemples d'utilisation de l'afficheur LCD

### 3.3.5 Programme principal

Grâce à la structuration du projet présentée précédemment, le programme principal devient particulièrement simple et lisible (voir ci-dessous). Ainsi, les étudiants peuvent comprendre aisément le principe qui sous-tend le fonctionnement interne d'un automate programmable.

```

void main(void)
{
    initialisation_maquette();
    Init_G7();
    do
    {
        lecture_entrees();
        calcul_receptivites();
        calcul_etat_futur();
        affectation_sorties();
        Etat_courant.etat =
            Etat_futur.etat;
    } while(1);
}

```

### 3.3.6 Synopsis

Le projet d'informatique industrielle qui vient d'être présenté constitue un excellent exercice de synthèse pour les étudiants. Il les sensibilise tout d'abord au rôle fondamental joué par la notion de machine séquentielle dans la réalisation de dispositif de contrôle/commande. La complexité du sujet, pourtant d'apparence simple, les oblige à structurer leur travail et leur programmation pour mener à bien le projet. Il démontre aussi concrètement l'intérêt d'utiliser les environnements d'édition GRAFCET proposés pour la programmation des automates programmables (Unity Pro<sup>®</sup> [6] dans notre département) par rapport à la démarche programmée de bas niveau. Il permet enfin de faire une large synthèse de bien des enseignements abordés depuis le début de l'année au niveau du microcontrôleur PIC, de sa programmation en langage C, et du langage GRAFCET (figure 8).

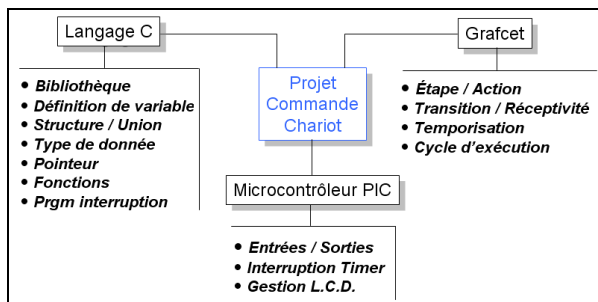


fig 8 : Synopsis du projet de commande d'un chariot

## 4 CONCLUSION

Bien que ce TP semble très simple au premier abord, la complexité du code développé (500 lignes et 17 fonctions) surprend les étudiants et les sensibilise à l'intérêt de faire appel aux outils intégrés de programmation Grafcet et aux Automates Programmables. Les étudiants sont aussi satisfaits de mener à bien un projet d'envergure mobilisant de nombreuses compétences.

La maquette servant de support à ce TP apporte une réponse concrète, performante et évolutive aux besoins pédagogiques précédemment exprimés.

Elle est utilisée dans la quasi-totalité des TP et des TD sur microcontrôleur PIC18F452.

Pour l'aspect matériel, l'ergonomie de la carte et le rapport performance/coût sont tels que de nombreux étudiants réalisent leur propre carte personnelle sur cette base pour développer leurs applications de projets tutorés et même de stage.

Pour l'aspect logiciel, les possibilités "classiques" de développement d'applications par organigramme en assembleur et en C sont complétées par l'approche d'implantations en GRAFCET (habituellement réservées aux automates industriels) ou en machines d'état (souvent cantonnées aux CPLD). Les étudiants découvrent ainsi toutes les possibilités de résolution de problèmes sur une plateforme unique et peuvent comparer en fonction de l'option initialement choisie :

- L'efficacité de l'approche,
- La pertinence du temps de développement,
- L'impact sur les ressources matérielles,
- Le résultat en fonctionnement réel.

Une maquette semblable destinée aux étudiants de seconde année et de licence professionnelle, disposant de connectivités USB et TCP/IP Ethernet, fera l'objet d'une prochaine publication.

## Bibliographie

- [1] Site département GEII de Montpellier : <http://iut.geii.montp2.free.fr>
- [2] Programme Pédagogique National du DUT "Génie Electrique et Informatique Industrielle", pp. 1 - , Septembre 2005. site Ministère de l'éducation , de l'enseignement supérieur et de la recherche, <http://media.education.gouv.fr/file/76/8/768.pdf>
- [3] Site Microchip : <http://www.microchip.com>
- [4] N. Bouteille, P. Brard, G. Colombari, N. Cotaina, D. Richet, "Le GRAFCET", Editions CEPADUES, 144 p, 1992.
- [5] Norme NF EN 60848 – Indice de classement C 03 890, "Langage de spécification GRAFCET pour diagrammes fonctionnels en séquence", Août 2002.
- [6] Site Schneider : <http://www.schneider-electric.fr>