Enseigner les microcontrôleurs en travaux pratiques : l'expérience IMA/Polytech'Lille

Blaise CONRARD, Anne-Lise GEHIN
Blaise.Conrard@polytech-lille.fr, Anne-Lise.Gehin@polytech-lille.fr
2, Bd Langevin, Bâtiment Polytech'Lille
59 655 Villeneuve d'Ascq cedex

RESUME : Cet article présente la plate-forme développée par le département IMA de Polytech'Lille pour les travaux pratiques d'enseignement des microcontrôleurs. Elle s'inscrit dans la première année de formation d'ingénieur dans le domaine de la conception de systèmes embarqués et ainsi propose la réalisation d'applications sur ce thème.

Cette plate-forme a été étudiée de manière à faciliter l'écriture des programmes, principale difficulté de l'enseignement des microprocesseurs et des microcontrôleurs. Les étudiants ne programment plus uniquement en assembleur, langage coûteux en temps de développement, de part sa complexité, des connaissances à acquérir et du peu de lisibilité des programmes développés qui deviennent, par ailleurs, vite longs. La solution proposée repose sur un langage beaucoup plus proche du langage naturel, des commandes en pseudo-langage se substituent à des sous-programmes assembleurs. Ce nouveau mode de programmation permet de dégager du temps pour se consacrer à l'étude des circuits d'entrées/sorties.

Par ailleurs, la plate-forme intègre un simulateur, capable de reproduire le comportement d'un microcontrôleur de type 68HC12, mais également celui d'un processus physique simple ou de diverses interfaces (claviers, afficheurs, boutons…). Ceci a permis d'élaborer une quinzaine de sujets très variés et très ludiques autours du thème des systèmes embarqués.

Enfin, la plate-forme présentée est actuellement entièrement logiciel et peut être ré-exploitée par d'autres établissements.

Mots clés : Microprocesseur, Travaux Pratiques, Assembleur, Compilateur, Simulateur.

1 INTRODUCTION

L'évolution des technologies, en terme de composants numériques programmables, confrontée au besoin croissant de fonctions attendues de plus en plus évoluées fait qu'actuellement presque tout équipement possède en son coeur un ou plusieurs composants de ce type.
Face à cette évolution, l'enseignement des microprocesseurs, microcontrôleurs ou plus généralement des circuits électroniques est un élément incontournable de la formation des ingénieurs en génie électrique. Ces enseignements s'articulent généralement en deux parties. La première à caractère théorique, réalisée sous la forme de cours et de travaux dirigés, cherche à faire comprendre le fonctionnement interne d'un microprocesseur et la façon de l'interconnecter avec des mémoires (RAM, ROM...) et des interfaces (port parallèle, port série, convertisseur analogique/numérique...) [1][2]. La seconde partie à caractère plus expérimentale, réalisée sous la forme de travaux pratiques, consiste généralement à étudier la programmation de ce type de composant [3]. Ceci se fait le plus souvent avec l'apprentissage de l'assembleur d'un microprocesseur ou d'un microcontrôleur particulier. Cet apprentissage, certes nécessaire, est très gourmand en temps car il nécessite d'apprendre un langage au jeu d'instructions assez riche, mais aussi à la structure relativement lourde et rigide contrairement à d'autres langages évolués tel que le C.

Cet article présente la plate-forme de TP développée au département IMA (Informatique, Microélectronique et Automatique) de l'école Polytech'Lille, destinée à l'enseignement des microprocesseurs. Cette plate-forme a été étudiée de manière à transmettre un maximum de compétences en un minimum de temps. Ainsi, un outil d'aide à la rédaction de programmes (éditeur de programmes) et un simulateur apte à faciliter leur debugage (ou leur déverminage) ont été réalisés. L'éditeur de programmes accélère l'écriture d'un code source par l'insertion dans le code d'instructions directement écrites en pseudo-langage. Ces instructions telles que des opérations arithmétiques entre variables ou des manipulations entre tableaux, se substituent à plusieurs instructions assembleur. La traduction d'une instruction écrite en pseudo-langage en assembleur reste néanmoins visible.
Le simulateur offre un environnement simple et ludique permettant de tester très rapidement les programmes réalisés en simulant éventuellement un processus physique associé à l'application étudiée.
L'article proposé s'articule en trois parties. La première, s'intéresse aux différentes formes de développement de programmes et à celle retenue. La seconde décrit les sujets de travaux pratiques et leur application sur un simulateur. La dernière fait un bilan de l'utilisation en enseignement de la plate-forme présentée et donne les informations nécessaires pour se procurer celle-ci.

Article publié par EDP Sciences et disponible sur le site http://www.j3ea.org ou http://dx.doi.org/10.1051/j3ea/2011016

2 DEVELOPPEMENT DE PROGRAMMES

Cette section s'intéresse aux différentes formes de développement de programmes, cherche à identifier leurs avantages et leurs faiblesses et s'achève sur la solution retenue au département IMA de Polytech'Lille.

2.1 Différents niveaux d'abstraction

Les travaux pratiques liés à l'enseignement des microprocesseurs se résument généralement au développement de programmes. Selon le niveau d'abstraction choisi, la rédaction de programmes peut se faire:

- en C (ou dans un langage évolué),
- en C intégrant des blocs rédigés en assembleur
- en assembleur avec la possibilité d'intégrer des commandes complexes nécessitant une pré compilation
- ou complètement en assembleur.

Le choix de l'utilisation d'un langage évolué permet une acquisition assez rapide de compétences pratiques dans la programmation des microprocesseurs. Cependant le fonctionnement interne d'un microprocesseur reste moins visible car le compilateur, par son travail de traduction en langage machine, décharge l'utilisateur d'un certain nombre de tâches telles que : la gestion des accumulateurs, la gestion de la pile et une partie de la gestion de la mémoire. En contre partie, l'activité de développement peut être plus rapide, l'étudiant pouvant d'avantage se concentrer sur la fonction à réaliser plus que sur la façon de la réaliser. Ce type de TP devient alors très intéressant pour l'étude de périphériques particuliers tels que le convertisseur analogique/numérique ou une interface réseau de type CAN (Control Area Network). L'article [4] présente une maquette orientée dans ce sens et utilisée à l'INSA de Toulouse.

Inversement, la rédaction de programmes uniquement en assembleur offre une meilleure compréhension des mécanismes de fonctionnement d'un microprocesseur, par l'illustration, entre autres, des mécanismes d'accès aux registres, de la gestion de la pile, de la gestion de la mémoire, de l'accès aux entrées/sorties. Mais ceci se fait au prix de séances de TP beaucoup plus longues. En effet, en raison de la rigueur de la syntaxe de ce langage, la longueur des programmes et leur difficile lisibilité, l'emploi de l'assembleur implique une rédaction plus lente et un nombre de cycles d'assemblage, de test puis de correction bien plus important. Enfin, ce type de programmation ne correspond plus aux pratiques de développement utilisées dans l'industrie et ne se présentera pas lors de la future carrière des étudiants.

En raison des faiblesses de chacune de ces approches, une solution intermédiaire a été choisie pour l'enseignement des microprocesseurs en travaux pratiques.

2.2 Du pseudo-langage dans l'assembleur

La solution retenue au département IMA de Polytech'Lille mixte assembleur et instructions en pseudo-langage afin de profiter des avantages de ces deux approches. L'idée est de travailler en assembleur mais d'autoriser l'insertion d'instructions complexes. Présenté dans CETSIS 2007 [5], un outil particulier a été développé. Son le rôle est de permettre l'écriture en pseudo-langage d'instructions complexes. Les instructions complexes sont alors automatiquement traduites en assembleur. Le code obtenu est inséré sous l'instruction complexe tandis que celle-ci est mise en commentaire. L'exemple ci-dessous illustre ce mécanisme de compilation où deux instructions complexes sont remplacées par le code assembleur correspondant :

Extrait du fichier initial :

```
        …
        ORG $8000
debut:  LDAA PortA
        CMPA #$80
        BNE suite
        var1<-var2+var3*2
        table[indice]<-var1
        …
```

Extrait du fichier après traitement :

```
        …
        ORG $8000
debut:  LDAA PortA
        CMPA #$80
        BNE suite

        ; var1<-var2+var3*2
        LDAA var3
        LSRA
        ADDA var2
        STAA var1

        ; table[indice]<-var1
        LDAA var1
        LDAB indice
        LDX #table
        STAA B,X
        …
```

Le fichier produit après ce prétraitement peut alors être assemblé de façon classique. Le résultat obtenu est alors le code machine implantable sur le microprocesseur ou sur un simulateur.

2.3 Intérêts et justification de ce choix

La solution retenue présente bon nombre d'avantages.

Elle permet aux étudiants d'écrire plus rapidement leurs programmes dans la mesure où une partie des opérations complexes n'a plus à être traduite en assembleur manuellement.

Elle conserve tous les intérêts pédagogiques d'un développement en assembleur, c'est-à-dire, entre autres, en constater les limites (instructions très élémentaires), gérer manuellement la mémoire (localiser les variables, la pile, le programme).

Dans la mesure où le code correspondant est affiché et remplace la ligne en pseudo-code dans le code assem-

bleur équivalent, l'aspect boite noire d'un compilateur traditionnel n'est pas présent.
Les mécanismes de transcodage d'une instruction complexes en langage assembleur et la non systématique optimalité du code généré permettent, en outre, à l'étudiant de comprendre le fonctionnement d'un compilateur et d'appréhender ses faiblesses.
Un autre avantage est que la solution proposée, incite à utiliser une démarche structurée qui consiste à préparer un algorithme avant de l'encoder. En effet, dans ce type de TP de programmation, l'attitude des étudiants est souvent de démarrer directement en assembleur et de développer au fur et à mesure leur algorithme, cette façon de faire étant souvent peu efficace.
Ainsi, plus globalement, la possibilité d'insérer des instructions complexes dans des programmes en assembleur forme une solution alliant rapidité d'utilisation et bonne compréhension du fonctionnement d'un microprocesseur.

3 MISE EN OEUVRE

3.1 Compréhension des interfaces

L'apprentissage du langage de programmation des microcontrôleurs ne représente qu'un aspect de l'enseignement de ces composants. En effet, l'intérêt majeur de ceux-ci est d'offrir une large variété d'entrées/sorties, facilitant leur mise en œuvre au sein de divers types d'applications. Plus spécifiquement, le microcontrôleur 68HC12 a été choisi pour ces travaux pratiques en raison du nombre d'interfaces qu'il offre : port parallèle, port série, timer, sorties MLI, port de communication pour réseau CAN, convertisseur analogique numérique…. Ces interfaces sont couramment utilisées pour le développement de systèmes embarqués.

Dans cet esprit, tous les sujets de travaux pratiques présentés proposent d'utiliser les interfaces suivantes, ou plus généralement une combinaison de celles-ci :

- port parallèle
- port série
- broche d'interruption
- convertisseur analogique/numérique
- timer pour des interruptions périodiques.

Cet aspect est important car il s'agit de montrer l'intérêt des interfaces qui déchargent le microcontrôleur de longs traitements de gestion des entrées/sorties au profit d'une activité de traitement des données de haut niveau et de coordination de ces interfaces.
D'un point de vue plus pratique, il s'agit également d'illustrer les aspects contrôle et communication entre le processeur et les interfaces. Ceci se fait au moyen de registres de commande ou d'état, complété par la possibilité de déclencher des interruptions dès qu'un évènement attendu se présente.

3.2 Choix des sujets de mise en application

La formation IMA de Polytech'Lille visant à former des ingénieurs dans le domaine des systèmes embarqués (appelés aussi parfois systèmes enfouis), les sujets de travaux pratiques ont été conçus dans ce sens. Ils permettent de développer diverses applications où le microcontrôleur doit interagir avec un environnement (utilisateur ou processus physique) via ses interfaces.

Répartie sur 5 groupes, la liste des sujets proposés est la suivante :

1) premier pas et ports parallèles :
   - chenillard,
   - détecteur de changement d'état,
   - serrure digitale pour coffre-fort,
2) ports parallèles, utilisation avancée :
   - compteur pour parking,
   - affichage sur afficheur 7-segments,
   - lecture d'un clavier matrice 4x4 16 touches,
3) communication sur liaison série :
   - barrière de parking contrôlée à distance par liaison série,
   - compteur de trafic routier communiquant,
   - affichage numérique contrôlé à distance,
4) utilisation des interruptions :
   - dé électronique,
   - minuterie d'un four (type micro-onde),
   - minuterie d'un four (version 2 avec multiplexage),
5) étude du convertisseur analogique/numérique :
   - capteur de température avec affichage digital,
   - régulateur de chauffage central,
   - radar routier.

L'ordre et le choix de ces sujets permettent d'utiliser graduellement toutes les interfaces disponibles. Les sujets sont conçus pour pouvoir réutiliser des éléments ou des parties de code d'un sujet à l'autre. L'emploi du traducteur pseudo-langage accélère le développement en permettant aux étudiants de se concentrer sur l'algorithme général. Globalement, 1h30 est le temps moyen consacré à la réalisation de chaque sujet, autrement dit une séance de 4h de TP permet de traiter 3 sujets.

Ce nombre de sujets, inhabituellement élevé pour ce type de TP, offre en fait plusieurs intérêts. D'une part, il permet, de parcourir largement les possibilités d'utilisation des microcontrôleurs. D'autre part, il s'adapte bien à ce type de travaux pratiques où on observe une grande disparité quant à la facilité des étudiants à programmer. Ainsi les plus forts ont la possibilité de montrer leur savoir faire sur des sujets ludiques et terminent les différents sujets proposés souvent bien avant la fin de la séance. Parallèlement, les plus faibles achèvent au moins le premier sujet, montrant ainsi qu'ils ont compris l'interface étudiée. Ceci évite la démotivation de ne jamais arriver au bout d'un TP (élément qui fut très préjudiciable à cet enseignement dans les années pré-

cédentes). Généralement, la grande majorité des étudiants traitent les sujets dans le temps imparti.

3.3 Plate-forme d'expérimentation

La plate-forme de TP ne comporte actuellement qu'un PC, sur lequel est implanté la chaîne de développement et le simulateur. L'objectif final est de compléter celle-ci par de véritables maquettes d'expérimentation qui sont actuellement en cours de réalisation.

La chaîne de développement requiert :

- un éditeur de texte : Crimson Editor © a été choisi pour sa facilité d'emploi (affichage simultané de plusieurs fichiers, numérotation des lignes, possibilité de lancer des applications externes)
- le traducteur pseudo-langage/assembleur : cette application "fait-maison" est présentée de façon détaillé dans [4]
- un assembleur : AS12 est utilisé.

Actuellement utilisée dans environnement Microsoft/Windows©, cette chaîne fonctionne également sous un environnement Linux. Elle permet la saisie des programmes et leur traitement jusqu'à obtenir un code machine.

En complément, le simulateur permet de tester les codes obtenus. Ce simulateur a été développé spécialement pour cette série de TP. D'autres solutions utilisant des logiciels commerciaux ou libres (SimHC12, NoICE for HC12 et les solutions libres GNU) ont été étudiées, mais aucune ne remplissait les exigences imposées. Les souhaits étaient d'avoir un simulateur :

- peu complexe, c'est-à-dire simple d'emploi, rapide et ne nécessitant pas des connaissances poussées en informatiques
- capable de gérer les interfaces du microcontrôleur ainsi que des processus physiques simple (évolution de la température d'une enceinte, circulation de véhicules…).

3.4 Le simulateur

Le simulateur développé offre les caractéristiques suivantes:

- simulation du fonctionnement d'un microcontrôleur 68HC12 (gestion de la mémoire, exécution des instructions, fonctionnement d'une partie des interfaces),
- simulation d'un processus physique simple et de composants externes au microcontrôleur (capteur, afficheur 7-segments, clavier, LED…).

Ce simulateur se veut simple d'emploi et offre une fenêtre graphique principale (figure 1) offrant les fonctionnalités suivantes:

- affichage de l'état du processeur (contenu des registres, valeur des drapeaux),
- affichage du contenu d'un bloc mémoire (pour une adresse, affichage hexadécimale des 16 octets consécutifs),
- un bouton de chargement d'un fichier de code machine,
- un bouton "Reset" permettant une réinitialisation du microcontrôleur,
- un bouton "Un pas" permettant l'exécution d'une unique instruction,
- un bouton "Go" (ou "Stop") permettant l'exécution continue du code (ou son arrêt),
- une ligne "Source", affichant l'instruction actuellement exécutée. Cette ligne est extraite du fichier résultant de l'assemblage et fournit son numéro, le code machine et le texte original du code source,
- un menu déroulant permettant de sélectionner le sujet de TP et d'obtenir l'interface utilisateur correspondante.

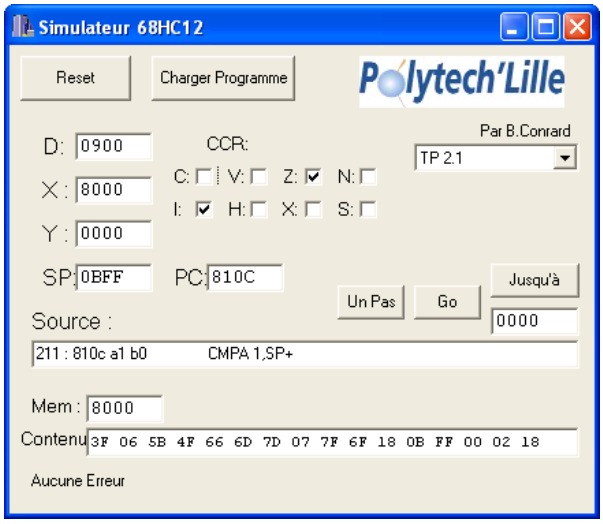

Figure 1 : Interface principale du simulateur IMA/Polytech'Lille

La sélection d'un sujet entraîne l'ouverture d'une seconde fenêtre dédiée au sujet sélectionné. Celle-ci fournit une interface graphique spécifique représentant le processus à commander et/ou l'interface utilisateur correspondante. Ces fenêtres se voulant très visuelles et très ludiques, un certain soin a été apporté à leur réalisation en offrant des boutons, des afficheurs, des véhicules en mouvement, quelques images… et indiquant par des étiquettes la broche ou le port du microprocesseur auquel chaque élément est connecté.

Les figures 2, 3 et 4 fournissent quelques exemples de fenêtres proposées :

Figure 2 :Interface du TP 1.3 : "Serrure digital d'un coffre"

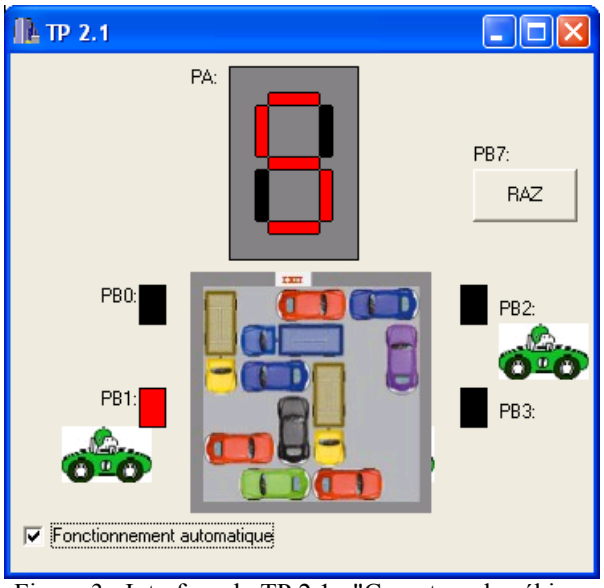

Figure 3 : Interface du TP 2.1 : "Compteur de véhicules pour parking"

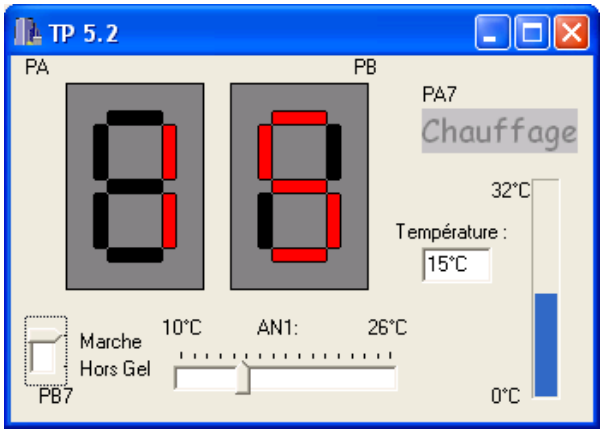

Figure 4 : Interface du TP 5.2 : "Régulateur de chauffage central"

Le détail du développement de ce simulateur dépasse le cadre de cette communication. Les principaux choix pour sa réalisation sont les suivants. Le simulateur est développé en C++ à l'aide de Borland C++ Builder 5 ©. Ce dernier a été choisi pour sa facilité à réaliser des interfaces graphiques et pour la rapidité du code obtenu.
La programmation d'objet a été fortement utilisée afin d'obtenir un code structuré, facile à faire évoluer et à maintenir. Des classes d'objet mémoires (RAM, ROM, E/S), microprocesseur, claviers, afficheurs ont été, ainsi, créées. Le jeu d'instructions du 68HC12 et les différents modes d'adressage associés ont été traduit en C++ ; seules quelques instructions complexes et non utilisées en TP n'ont pas été implantées. Le mécanisme de gestion des interruptions a été intégré aussi bien pour le processeur que pour les interfaces d'E/S. Sur des machines récentes, la vitesse du simulateur est proche de celle du microcontrôleur réel.

4 BILAN ET RETOUR D'EXPERIENCE

Le retour d'expérience après une année de test de l'utilisation des outils présentés est le suivant.
Les étudiants acquièrent en 5 séances de 4h de TP, une compréhension :
- du fonctionnement d'un microprocesseur,
- des mécanismes de compilation,
- de l'intérêt des interfaces d'entrées/sortie (décharger le microprocesseur des activités basiques pour faire du traitement de données de plus haut niveau). Ils savent également mettre en œuvre ses interfaces,
- des interruptions. Ceci est un prérequis pour aborder ultérieurement les systèmes multi-tâches et/ou temps réel,
- de ce qu'est un système embarqué ou enfoui, c'est-à-dire un système informatique programmé complètement intégré au système qu'il contrôle.

Pour aboutir à ce résultat, le département IMA a développé avec peu de moyens financiers (mais avec beaucoup de temps) :

- un pré-compilateur chargé de traduire des instructions en pseudo-langage dans un code assembleur au contenu visible et vérifiable par l'utilisateur,
- un simulateur du microprocesseur 68HC12 associé à 15 applications différentes permettant la mise en œuvre de nombreuses interfaces (port parallèle, port série, interruption, convertisseur analogique/numérique, timer).

A l'issu de cette première année d'utilisation, l'équipe enseignante est très satisfaite des résultats obtenus. Les résultats obtenus par les étudiants lors des examens ont montré que la grande majorité d'entre eux a acquis les compétences souhaitées et ce dans une proportion supérieure à celle des années antérieures. De même, le rejet pour la programmation assembleur, très marqué les années précédentes, a presque disparu. Du point de vue des étudiants, ces TP ont reçu un très bon accueil.

Ils ont apprécié d'une part la progression linaire de la difficulté des problèmes posés, et d'autre part, l'aspect ludique des sujets et du simulateur. Les étudiants issus d'IUT, ayant déjà suivi une formation équivalente, ont reconnu que ces TP étaient plus intéressants et allaient plus loin dans la compréhension et dans la maîtrise des microcontrôleurs. Un autre point très apprécié des étudiants est la possibilité de télécharger les logiciels afin de s'exercer sur leur machine personnelle.

Malgré cela, les enseignants (mais pas les étudiants) ont regretté l'absence de maquettes réelles permettant la mise en œuvre finale des programmes développés. Un projet a été lancé dans ce sens et un prototype a été conçu et réalisé également au sein de l'école. Cette maquette (cf. figure 5) offre 3 afficheurs 7-segments multiplexé, des LED, un clavier matriciel 12 touches, 4 interrupteurs, un potentiomètre connecté au convertisseur analogique/numérique, 2 boutons permettant de générer des interruptions de type IRQ et XIRQ et 2 ports série de type RS232. Cette variété d'interfaces offrent aux étudiants la possibilité de tester sur un matériel réel l'emploi des entrées/sorties, vues actuellement uniquement en simulation. D'ici quelques mois (prévu pour mars 2010), le simulateur sera enrichi d'une nouvelle interface simulant cette maquette de sorte à permettre aux étudiants de tester et corriger leurs programmes avant de les faire exécuter directement par la maquette. Cette nouvelle maquette va être utilisée dès cette année, la production de 7 autres cartes devant être achevée pour le début des prochaines séances de travaux pratiques.

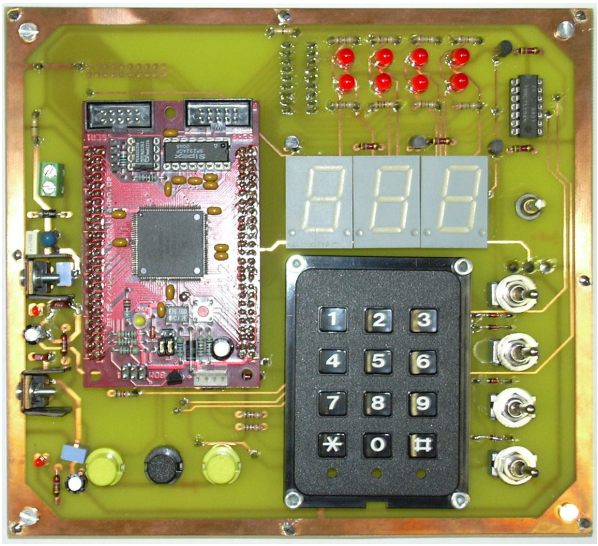

Figure 5 : Maquette de programmation à base de 68HC12

Afin d'être librement réutilisée dans d'autres établissements d'enseignement ou à des fins d'auto-formation individuelle, l'adresse Internet suivante regroupe l'ensemble des outils logiciels nécessaires à la mise en place ou à l'évaluation de cette plate-forme de TP, ainsi que les énoncés des sujets :

http://ima1.polytech-lille.net/TPmP

5 CONCLUSION

Cet article présente la plate-forme utilisée pour les travaux pratiques en microprocesseur/microcontrôleur du département IMA (Informatique Microélectronique Automatique) de Polytech'Lille. Libre d'utilisation par d'autres établissements d'enseignement, cette plate-forme contient un outil d'aide à la rédaction rapide de programme assembleur et un simulateur, capable de reproduire l'utilisation d'un microcontrôleur de type 68HC12 pour différentes applications embarquées.
Utilisée depuis un an à Polytech'Lille par des étudiants en 1ère année du cycle ingénieur, cette plate-forme a montré son efficacité en terme d'heures d'enseignement nécessaires, pour l'acquisition des principes et des concepts de base des microcontrôleurs et de l'informatique embarqué.

Bibliographie

[1] G. Gateau, J. Regnier et J.C. Hapiot, Etude du principe de fonctionnement d'une unite centrale de traitement CPU, Actes de CETSIS 03, p173, Toulouse, 2003

[2] S. Reboul, J.-B. Choquel, "Synthèse d'un microprocesseur en langage VHDL", Actes de CETSIS 05, Nancy, 2005

[3] V. Mahout, F. Pompignac, J. Martin, B. Faure et P. Belaubre, Du langage d'assemblage à la commande numérique de processus, Actes de CETSIS 03, p255, Toulouse, 2003

[4] S. Bouter, R. Malti, "Mise en œuvre d'un bus de terrain CAN à partir de microcontrôleur HC12", Actes de CETSIS 07, Bordeaux, 2007

[5] B. Conrard, A.-L. Gehin, "Du pseudo-langage dans l'assembleur", Actes de CETSIS 07, Bordeaux, 2007