

La liaison série asynchrone étudiée dans le cas d'un lecteur de carte à puce

Vincent Creuze, Isabelle Pinon, Didier Crestani, Jean-Paul Delauzun, Bertrand Gélis, Pierre Gillet, Eric Maurines, Eric Pommier, Xavier-François Hochmuth

▶ **To cite this version:**

Vincent Creuze, Isabelle Pinon, Didier Crestani, Jean-Paul Delauzun, Bertrand Gélis, et al.. La liaison série asynchrone étudiée dans le cas d'un lecteur de carte à puce. CETSIS: Colloque sur l'Enseignement des Technologies et des Sciences de l'Information et des Systèmes, Oct 2011, Trois-Rivières, Canada. pp.001-006. lirmm-00638742

HAL Id: lirmm-00638742
https://hal-lirmm.ccsd.cnrs.fr/lirmm-00638742

Submitted on 7 Nov 2011

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.

La liaison série asynchrone étudiée dans le cas d'un lecteur de carte à puce

Vincent Creuze, Isabelle Pinon, Didier Crestani, Jean-Paul Delauzun, Bertrand Gélis,
Pierre Gillet, Eric Maurines, Eric Pommier et Xavier-François Hochmuth
Vincent.creuze@univ-montp2.fr
IUT de Montpellier – Département GEII (Génie Electrique et Informatique Industrielle)
99 avenue d'Occitanie, 34 296 Montpellier Cedex 5, France

RESUME : Dans cet article, nous présentons une approche expérimentale de la liaison série asynchrone pour des étudiants de première année d'IUT GEII (Génie Electrique et Informatique Industrielle). Au cours de la séance de Travaux Pratiques que nous détaillons ici (module II2 du PPN 2005, Architecture de Systèmes à Processeurs), nous utilisons la liaison série (USART, Universal Synchronous Asynchronous Receiver) d'un microcontrôleur PIC 18F452 pour lire des données issues d'une carte à puce. Ce TP ouvrant de nombreuses possibilités d'approfondissements (protocole utilisé, types d'accès…), il peut être utilisé en tant qu'introduction à la liaison série aussi bien en IUT qu'à des niveaux supérieurs. La présentation de ce TP est complétée par les résultats d'un sondage révélant comment les étudiants ont perçu cette façon d'aborder la liaison série.

Mots clés : Informatique industrielle, microcontrôleur, liaison série, carte à puce, retour d'expérience.

1 INTRODUCTION

Parmi les nombreuses notions d'informatique industrielle inscrites au Programme Pédagogique National (PPN) du Diplôme Universitaire de Technologie (DUT) en Génie Electrique et Informatique Industrielle (GEII) [1], on trouve l'étude des microcontrôleurs et de leurs différentes fonctionnalités. A l'Institut Universitaire de Technologie de Montpellier (IUT), au département GEII nous avons réparti cet enseignement sur les deux années de formation. En première année, l'apprentissage du microcontrôleur se fait sous forme de cours, de Travaux Dirigés (TD), de Travaux Pratiques (TP) et d'Etude et Réalisation (ER). Les étudiants acquièrent ainsi les bases qu'ils exploiteront ensuite en deuxième année dans un enseignement davantage orienté « projet » (Etude et Réalisation, participation à des concours de robotique).

Au terme de la première année, les étudiants ont ainsi abordé de nombreuses fonctionnalités d'un microcontrôleur PIC™ en langage C (compilateur C18 de Microchip™) et dans une moindre mesure en assembleur. Parmi celles-ci, on trouve la génération d'un signal PWM, la communication avec un afficheur à cristaux liquides (LCD), mais aussi l'utilisation des interruptions, des timers et des convertisseurs. La dernière notion abordée est la liaison série (USART). Afin de rendre attrayant l'apprentissage du fonctionnement du PIC, nous tâchons autant que possible d'associer les fonctionnalités étudiées à des applications concrètes ou ludiques. Pour la liaison série, nous proposons un TP consistant à recevoir et enregistrer la première trame émise par une carte à puce (à contacts) lors de sa mise sous tension.

Dans cet article, nous nous proposons de décrire le contenu de ce TP et la façon dont les étudiants le perçoivent. Cet article est composé de quatre parties. La première présente le contexte et les objectifs du TP. Nous décrivons ensuite le matériel utilisé (lecteur de carte à puce et maquette d'étude du microcontrôleur PIC). Dans la troisième partie nous exposons les principales étapes du TP et, pour chacune d'entre elles, nous proposons un court exemple de code en langage C. Enfin, nous analysons les résultats d'un sondage mené auprès d'une centaine d'étudiants à l'issue de cette séance de Travaux Pratiques et nous présentons la plate-forme pédagogique ayant permis d'effectuer ce sondage.

2 OBJECTIFS DU TP

Il existe deux familles de cartes à puce : les cartes à mémoire (télécartes, cartes d'accès…) et les cartes à microcontrôleur (cartes bancaires, cartes vitales ou javacards, cartes SIM…). Dans ce TP, nous étudions le cas le plus courant, c'est-à-dire la carte à microcontrôleur. Il existe plusieurs technologies très différentes parmi les cartes à microcontrôleur. Malgré cela, la majorité de celles-ci respectent les normes ISO 7186 (dimensions de la carte et positionnement de la puce) et ISO 7816 (emplacement des contacts, signaux électroniques et protocoles de transmission). L'échange de données (lecture ou écriture) avec la carte se fait par une seule broche (I/O). La carte et le lecteur de carte émettent donc chacun à leur tour des bits de données en série (liaison série « half-duplex »). Il existe plusieurs vitesses et plusieurs protocoles différents pour l'échange des données. Lorsqu'on introduit une carte à puce dans un lecteur, il est donc nécessaire que la carte indique à celui-ci à quelle vitesse et selon quel protocole elle souhaite communiquer. Ces premières données indispensables forment une trame d'octets, nommée ATR (Answer To Reset) et émise automatiquement par la carte au démarrage de la communication. Des informations plus détaillées sur les cartes à puce et leur fonctionnement peuvent être trouvées dans l'excellent livre de Christian Tavernier [2].

Dans le TP que nous présentons ici, nous connectons la broche I/O à l'entrée série d'un microcontrôleur PIC18F452. Nous initialisons la carte à puce (procédure de RESET). Puis, au moyen du port série du PIC, nous enregistrons, visualisons et analysons la trame

d'ATR émise par la carte. Enfin, nous vérifions si la trame reçue n'est pas entachée d'erreurs de parité. La séance de TP dure trois heures.

3 MATERIEL UTILISE

3.1 La carte à puce

Afin de pouvoir multiplier les scenarii (divers ATR, transmission d'octets avec simulation d'erreur de parité,..) nous avons choisi d'utiliser une carte à puce vierge de type Gold (ou Wafer 1). Elle contient un microcontrôleur PIC 16F84 et une mémoire EEPROM série 24LC16 de 2Ko. Nous avons utilisé le logiciel gratuit Microchip™ MPLAB IDE™ [3] pour écrire le programme embarqué dans la carte et générer le fichier hex. Ce dernier est ensuite transféré dans l'EEPROM à l'aide d'un quelconque lecteur/programmateur de carte à puce connecté à un PC. Dans notre cas, le programme a été rédigé en assembleur, de façon à garantir la vitesse de transmission des données. En effet, le PIC 16F84 ne dispose pas d'USART et il est nécessaire d'émuler la liaison série sur un port standard. Comme toutes les cartes à puce, la carte Gold présente une zone de contacts (broches) répartis comme indiqué sur la figure 1.

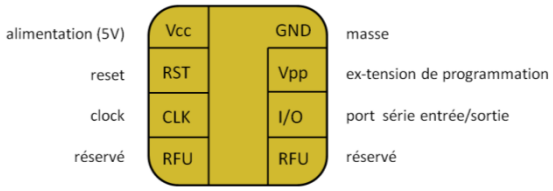


fig 1 : Broches (ou contacts) d'une carte à puce.

3.2 Le lecteur de carte à puce et le microcontrôleur

Au département GEII de l'IUT de Montpellier, nous avons conçu une maquette pédagogique composée d'un programmateur de PIC via USB et d'un microcontrôleur PIC 18F452 environné par de nombreux accessoires [4] (Fig. 2) : interrupteurs, diodes électroluminescentes, afficheur à cristaux liquides, connecteurs BNC pour entrées analogiques, capteur de température I2C et également un connecteur DB25 permettant le raccordement de nouvelles cartes d'extension.
Afin de relier la broche I/O de la carte à puce à l'entrée de l'USART du PIC de la maquette principale, nous avons conçu une maquette d'extension (Fig. 3). Celle-ci comporte un support avec connecteur pour carte à puce. Lorsqu'on insère la carte, les contacts GND et VCC de la carte sont automatiquement alimentés. La maquette envoie alors automatiquement une horloge de fréquence 3.579 MHz sur la broche CLK de la carte à puce. Les autres contacts de la carte à puce sont reliés au PORTC du PIC18F452. En particulier, la broche I/O de la carte à puce est reliée à l'entrée RC7 du PIC, correspondant au RX de l'USART et donc dédiée à la liaison série. La broche RST est quant à elle connectée à l'entrée RC5 (Fig. 4). Enfin, un bit nommé *insertion* correspond à un contacteur mécanique qui est à 1 si la carte est présente dans le lecteur et à 0 sinon. Il sera utilisé par les étudiants pour déclencher le reset et la lecture de la carte.

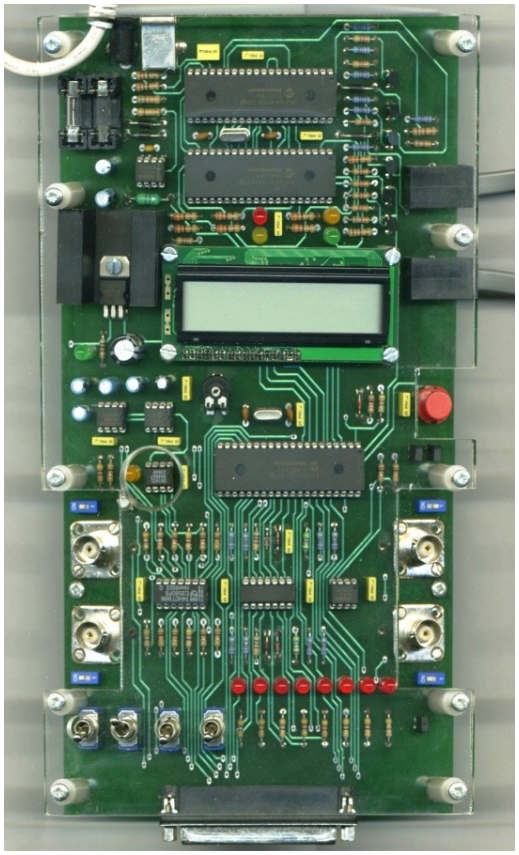


fig 2 : Maquette d'étude du PIC 18F452

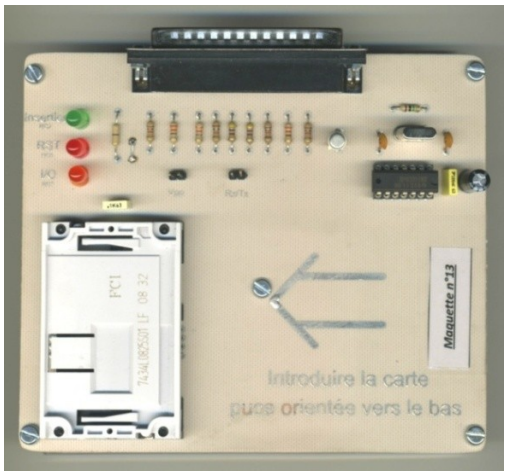


fig 3 : Plaquette permettant de connecter la carte à puce au microcontrôleur PIC. En bas à gauche, on voit le connecteur de carte à puce. En haut à droite, le circuit générant l'horloge fournie à la carte à puce. En haut, le connecteur noir permet de connecter cette plaquette à la carte mère présentée sur la figure 2

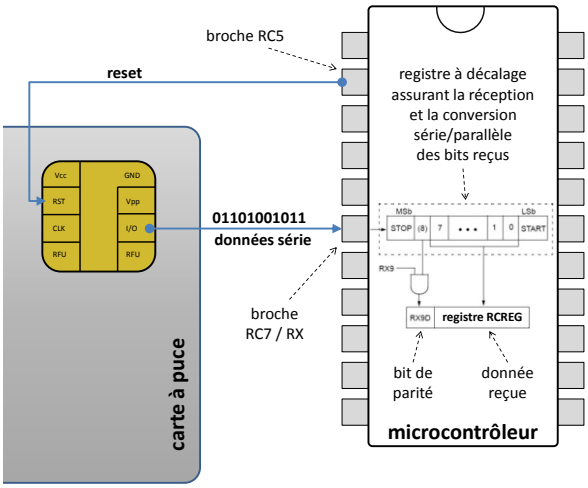


fig 4 : Connexions entre la carte à puce et le microcontrôleur et mise en œuvre interne de la conversion série/parallèle

4 CONTENU DU TP

4.1 Configuration de l'USART du PIC

La broche de données de la carte à puce est reliée à la broche RC7 du PIC18F452. Il s'agit de la broche qui correspond à l'entrée RX du port série du PIC18F452. Les étudiants doivent donc configurer l'USART du PIC pour recevoir et enregistrer les octets de l'ATR (au préalable ils ont reçu un cours consacré à l'utilisation de l'USART du PIC dans le cadre du module II2 [1]).
La première trame émise par une carte à puce (ATR, Answer To Reset) respecte systématiquement le débit d'émission suivant :

$$Débit\ de\ la\ trame\ ATR = \frac{f_{CLK}}{372}\ bits/s$$

Puisque la maquette fournit à la carte un signal d'horloge à 3.579MHz, les étudiants calculent le débit de la trame ATR, qui est égal à 9.620 bits/s. Ils en déduisent les valeurs optimales de SPBRG et BRGH, les paramètres de réglage du débit de l'USART du PIC. Ensuite, on explique aux étudiants que chaque octet émis par la carte à puce débute par un bit de start (=0) et se termine par un bit de parité paire (« even » en anglais) et deux bits de stop (=1), également appelés temps de garde (TG). Le module USART du PIC est conçu pour fonctionner avec ce format de données. Cependant, s'il permet d'enregistrer le bit de parité, il ne permet pas de vérifier automatiquement cette dernière (cela fera l'objet de la dernière partie du TP). A ce stade, on suppose que les données reçues sont bonnes et on ignore la parité.
La configuration de l'USART s'effectue donc de la façon suivante (compilateur C18 de Microchip™) :

```
//liaison série asynchrone, BRGH=1
TXSTA = 0x04 ;

//Réglage vitesse de la liaison série
SPBRG = 64 ;

//Allumage USART, mode 9 bits (9 bits
+ parité stockée à part dans RX9D)
RCSTA = 0b11000000;
```

4.2 Reset de la carte à puce

Dès que l'on introduit la carte à puce dans le lecteur, il faut procéder au reset de la carte en respectant les contraintes définies par la norme :

- Le bit RST doit être initialement à 0.
- Une fois que la carte est introduite, on doit maintenir RST à 0 pendant 40.000 coups d'horloge CLK.
- Enfin, on met RST à 1 et la carte à puce envoie son ATR après un délai compris entre 400 et 40.000 coups d'horloge.

La mise en œuvre du reset de la carte à puce peut être programmée de la façon suivante :

```
#define reset_carte     PORTCbits.RC5
#define insertion  PORTCbits.RC2

reset_carte = 0;

//attente insertion carte
while (insertion == 0);

//Attente de 11,17ms (40 000*T_CLK)
Delay1KTCYx(28);

//Fin d'application du reset
reset_carte = 1;
```

4.3 Enregistrement de l'ATR

Peu de temps après la fin du reset, la carte à puce émet la trame d'ATR. Nous demandons donc aux étudiants de stocker les octets émis dans un tableau nommé ATR[ ]. L'algorithme est le suivant (on rappelle que lorsqu'un octet est reçu par l'USART, le bit RCIF passe à 1 et l'octet est lisible dans le registre RCREG) :

- i = 0
- Autorisation de la réception série (bit CREN)
- Faire :
  - attendre que RCIF passe à 1
  - Enregistrer RCREG dans ATR[i]
  - incrémenter i

  Tant que RCREG ≠ 0.
- Arrêt de l'autorisation de réception série

Une fois l'ATR enregistré par le PIC 18F452, les étudiants vont lire les valeurs stockées dans la RAM du PIC à l'aide de la fonction « Watch » du logiciel MPLAB. Les plus performants peuvent aussi afficher l'ATR sur l'afficheur à cristaux liquides (LCD) de la

maquette (une bibliothèque de fonctions d'affichage est à leur disposition pour ce TP).
Le programme correspondant à l'algorithme ci-dessus peut être écrit de la façon suivante :

```
unsigned char ATR[10], i = 0 ;

RCSTAbits.CREN = 1 ;    //Enable RX
do
     {    while (PIR1bits.RCIF == 0);
          ATR[i] = RCREG;
     } while (ATR[i++] != 0);
RCSTAbits.CREN = 0 ;    //Disable RX
```

4.4 Interprétation de l'ATR

La trame d'ATR comporte un nombre variable de caractères (octets), envoyés dans l'ordre suivant **: TS ; T0 ; TA1 ; TB1 ; TC1 ; TD1 ; TA2 ; TB2 ; TC2 ; TD2 ; TA3… ; …octets d'historique… ; TCK.**
Seul le premier caractère **TS** est obligatoire, tous les autres sont facultatifs. Chaque caractère a une signification particulière, brièvement rappelée ci-dessous :
TS : Caractère initial de l'ATR, indiquant la **convention** utilisée par la carte (0x3F=convention inverse, 0x3B=convention directe). La convention utilisée définit l'ordre dans lequel les bits d'un octet sont envoyés et quels niveaux de tensions correspondent à 1 et 0. Il se trouve que notre carte et le PIC 18F452 utilisent la même convention. Nous n'aurons donc pas à nous soucier de ce caractère, mais il sera intéressant de le décoder pour savoir de quelle convention il s'agit.
T0 : Caractère de **format** de l'ATR, indiquant les bits contenus dans la trame ATR (ex : 0x10 signifie que l'ATR ne contient que TA1 et qu'il n'y aura pas de caractères d'historique).
TA1 : Caractère donnant les paramètres D et F permettant de calculer la **vitesse de transmission** de la carte.
TB1 : Caractère anciennement utilisé pour définir la **tension de programmation** VPP. L'évolution de la technologie a rendu cette broche inutile. Elle est donc soit connectée au +5v, soit déconnectée.
TC1 : Caractère rarement utilisé définissant un **temps de garde supplémentaire** (durée d'attente supplémentaire entre les octets lors des transmissions).
TD1 : Caractère indiquant la présence éventuelle de caractères TA2, TB2, TC2, et TD2. Ce caractère indique aussi le protocole d'échange de données utilisé par la carte (protocole T0, protocole T1 ou autres protocoles). Seuls les protocoles T0 et T1 sont clairement définis par la norme. Parmi eux, le protocole T0 est le plus utilisé.
TCK : Ne doit être présent que si un protocole différent de 0 a été spécifié. Dans le cas où il est présent, c'est un XOR de tous les octets précédents.

Remarque : L'existence des caractères d'historique (16 au maximum) est définie par la norme, mais ni leur contenu, ni leur finalité ne sont précisés. Autrement dit, ils sont librement utilisables par le concepteur de la carte. Souvent ils servent à identifier la carte. Encore plus souvent, ils ne sont pas utilisés.

Les étudiants disposent de tous les documents permettant de décoder l'ATR. Les valeurs programmées sur notre carte à puce sont **: 0x3B ; 0x90 ; 0x11 ; 0x00**. Ce qui signifie que la carte utilise la convention directe (d'après le premier caractère, TS = 0x3B), que l'ATR contient les caractères TA1 et TD1 et ne contient pas de caractère d'historique (d'après le deuxième caractère, T0 = 0x90), que la vitesse de transmission sera égale à $f_{CLK}/372$ (d'après le troisième caractère, TA1 = 0x11) et enfin qu'il n'y aura pas de caractères TA2, TB2, TC2 et TD2 (d'après le dernier caractère, TD1 = 0x00).

4.5 Détection d'erreurs de parité

Chaque octet transmis par la carte est accompagné d'un bit de parité permettant de vérifier que la transmission s'est déroulée sans erreur. Ce bit de parité correspond à un « ou exclusif » des 8 bits de l'octet.
Les cartes à puces de l'IUT sont volontairement programmées pour produire une erreur de parité sur certains caractères de l'ATR.
Une fois que le programme de lecture de l'ATR fonctionne, on demande donc aux étudiants de compléter leur programme afin que la parité soit vérifiée pour chaque caractère reçu (le bit de parité reçu se trouve dans RCSTAbits.RX9D). Dans le cas où la parité est mauvaise, le caractère 0x99 est enregistré dans le tableau ATR[i] à la place du caractère reçu. Le module USART du PIC ne comporte pas de calcul automatique de parité. Il faut donc la calculer, par exemple de la façon suivante :

```
unsigned char octet_recu ;
unsigned char parite_calculee ;

octet_recu = RCREG;
parite_calculee     =     (octet_recu
^(octet_recu>>1)^(octet_recu>>2)^
(octet_recu>>3)^(octet_recu>>4)^
(octet_recu>>5)^(octet_recu>>6)^
(octet_recu>>7))&0x01;

if (parite == RCSTAbits.RX9D)
          ATR[i++] = octet_recu;
else      ATR[i++] = 0x99;
```

5 RETOUR D'EXPERIENCE

5.1 Outil de sondage

A la fin de la séance de TP, nous avons réalisé un sondage anonyme afin de recueillir l'avis des étudiants (une centaine) sur cette façon d'aborder la liaison série. Afin de simplifier cette démarche, avec le concours de la Direction des Usages du Numérique de l'Université Montpellier 2, nous avons mis en place le sondage au moyen de la **plate-forme pédagogique Claroline**. C'est une plate-forme Open Source, distribuée sous licence GPL, qui permet à des centaines d'institutions de créer gratuitement des espaces cours en ligne [5]. Pour chaque cours, le formateur dispose d'une série d'outils (parcours pédagogiques, cours, exercices, sondages, annonces…).

5.2 Résultats du sondage

Les graphes de la figure 5 présentent les questions posées et les résultats obtenus.
Concernant la perception de ce TP par les étudiants, on constate que, très majoritairement (88%), ils ont apprécié l'utilisation d'une application de la vie courante pour aborder une notion qui leur paraissait abstraite au départ (la liaison série asynchrone). Qui plus est, ce type de TP renforce leur intérêt pour l'informatique industrielle en mettant en évidence la multitude et l'utilité de ses applications quotidiennes. Ainsi à l'issue de ce TP, 88% des étudiants trouvent que l'informatique industrielle est utile, voire très utile, alors que peu d'entre eux connaissaient cette discipline en entrant à l'IUT.
Concernant l'efficacité du TP, on remarque que les étudiants ont l'impression d'avoir augmenté leurs compétences lors de cette séance (82%).
Enfin, en ce qui concerne le dimensionnement du TP, on constate que plus de la moitié des étudiants (54%) trouve ce dernier trop long par rapport au temps imparti (trois heures). Nous avons constaté en effet, lors des séances de TP, qu'une proportion équivalente d'étudiants n'avait pas eu le temps d'aborder la vérification de la parité. Ce problème a été pris en compte pour l'année prochaine de la façon suivante. La description détaillée du contenu de l'ATR occupe une grosse partie du TP et constitue la principale source de difficultés et d'interrogations des étudiants, alors qu'il ne s'agit que d'une notion secondaire. Afin de concentrer le travail demandé sur l'utilisation de l'USART et la prise en compte de la parité, nous avons décidé de placer désormais l'interprétation de l'ATR en annexe du TP. Cela souligne le caractère accessoire de cette question et rassure les étudiants qui apprécient de pouvoir terminer entièrement les TP (hors annexes). De plus, cela leur permet de distinguer plus facilement les notions essentielles (qui méritent d'être retenues), des notions secondaires. Nous avons choisi aussi de ne pas supprimer complètement cette question car elle permet aux étudiants les plus rapides ou les plus intéressés

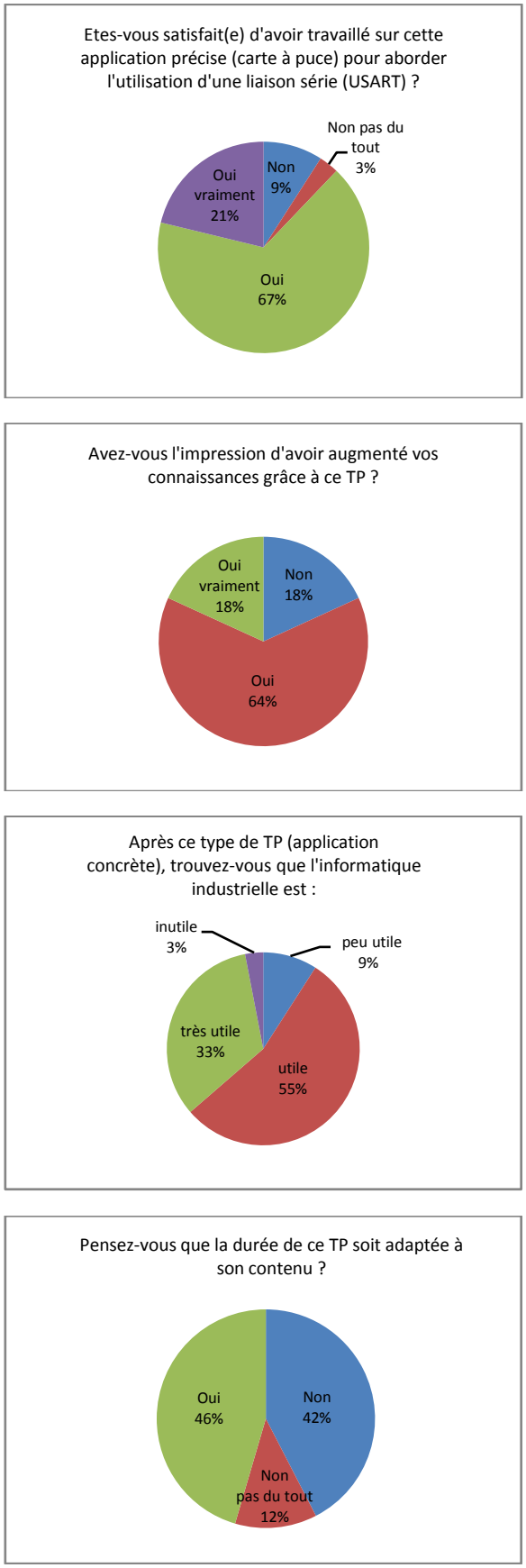


Fig 5 : Résultats du sondage réalisé à l'issue du TP

(éventuellement après la séance) d'approfondir leur compréhension du fonctionnement de la carte à puce. Cependant, ce que les étudiants doivent absolument retenir de ce TP, c'est que le microcontrôleur reçoit et stocke dans un registre les bits de la liaison série de façon asynchrone par rapport au déroulement du programme principal. La fin de la réception d'un octet est marquée par une interruption ou un drapeau et l'on peut alors aller lire la donnée dans le registre de réception.

6 CONCLUSION

Nous avons présenté une séance de travaux pratiques consacrée à l'étude de la liaison série d'un microcontrôleur PIC. Pour cela nous avons utilisé l'exemple d'une communication avec une carte à puce. Le déroulement des séances de TP ainsi que le sondage réalisé auprès des étudiants montrent que des exemples aussi concrets sont très appréciés et augmentent l'intérêt des étudiants pour l'informatique industrielle. Ce sondage a également mis en avant un problème de dimensionnement du TP que nous avons pris en compte pour l'année prochaine. La nouvelle version de ce TP devrait concentrer le travail des étudiants sur les notions essentielles, tout en permettant l'approfondissement pour ceux qui le désirent.

Le sondage a été l'occasion pour notre équipe pédagogique d'une première utilisation de la plate-forme Claroline. Cela nous a permis de mesurer son efficacité et sa simplicité d'utilisation et nous songeons désormais à intensifier son usage. Dès la rentrée prochaine, nous projetons ainsi de l'utiliser pour compléter la formation et l'évaluation en automatique pour les étudiants de deuxième année de l'IUT GEII de Montpellier.

Bibliographie

[1] Programme Pédagogique National du DUT "Génie Electrique et Informatique Industrielle", Septembre 2005, site du Ministère de l'éducation, de l'enseignement supérieur et de la recherche (France) : http://media.education.gouv.fr/file/76/8/768.pdf

[2] C. Tavernier, « Les cartes à puce : Théorie et mise en œuvre », *Dunod Editeur, 2ème édition, décembre 2006*.

[3] Site Microchip : http://www.microchip.com

[4] D. Crestani, V. Creuze, X.F. Hochmuth, E. Maurines, E. Pommier, « Un projet fédérateur d'Informatique Industrielle en IUT GEII : Programmation d'un GRAFCET en langage C sur PIC 18F452 », *Actes du 8ème Colloque sur l'Enseignement des Technologies et des Sciences de l'Information et des Systèmes (CETSIS 2010), Grenoble (France), mars 2010.*

[5] Site web de la plate-forme pédagogique Claroline : http://www.claroline.net/