IFT 1227
Architecture des ordinateurs I

Jean Pierre DAVID

Démo 8

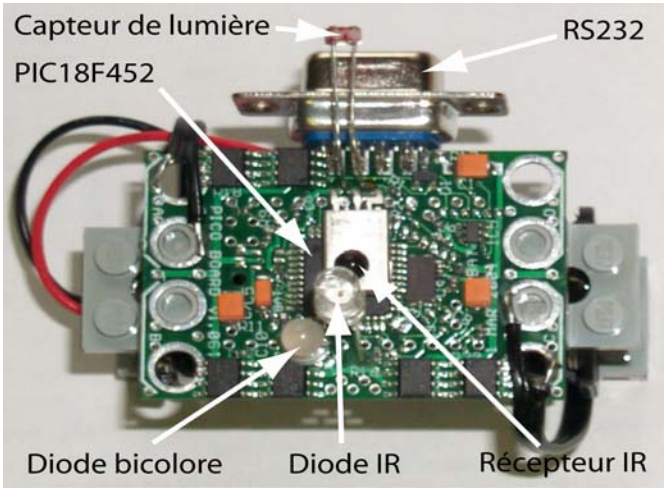
A. Installation

- Décomprimez « david.zip » dans le répertoire c:\cygwin\home
- Décomprimez « labo8.zip » dans le répertoire r:
- Double-cliquez sur le fichier r:*Config MPLAB.reg* et confirmez.

B. Introduction

Nous avons conçu la carte "Pico-board". Celle-ci contient un microcontrôleur PIC18F452 que l'on peut reprogrammer à volonté avec un PC au moyen du port série ou directement par une communication Infra-Rouge (IR) semblable à celles utilisées dans les télécommandes domestiques. La carte est dotée de :

- ✓ Une diode IR (pour émettre des messages).
- ✓ Un récepteur IR (pour recevoir les messages).
- ✓ Une diode bicolore rouge/vert pour transmettre une information visuelle.
- ✓ Un capteur de lumière.
- ✓ Des interfaces de puissance pour actionner des moteurs ou d'autres périphériques.



Cette carte est montée sur un petit robot LEGO constitué essentiellement de deux roues motorisées et d'une armature pour les supporter. Dans ce travail vous avez à écrire des petits programmes en assembleur pour contrôler les divers périphériques de la carte. Un certain nombre de fonctionnalités ont déjà été préprogrammées. Elles constituent le « firmware », une sorte de système d'exploitation qui permet aux développeurs un accès plus facile aux ressources du microcontrôleur et à ses périphériques.

Le « firmware » gère principalement :

- ✓ Une horloge temps réel.
- ✓ L'envoi et la réception de messages IR.
- ✓ La programmation par IR.

- ✓ La commande de puissance et de direction des moteurs.
- ✓ La mesure de l'intensité lumineuse.

Le développeur doit donc faire appel aux routines du « firmware » chaque fois qu'il désire utiliser une de ces fonctionnalités. La diode bicolore n'est pas gérée par le « firmware » et est directement connectée aux broches RB4 et RB5 du microcontrôleur. Une diode bicolore a trois modes de fonctionnement :

1) Aucun courant ne passe : elle est éteinte
2) Le courant passe dans un sens : elle est verte
3) Le courant passe dans l'autre sens : elle est rouge

Pour faire passer le courant dans un sens ou dans l'autre, on joue avec la « pression électrique » sur chacune de ses broches. Pour vous donner un point de départ, nous vous fournissons un petit programme « led.asm » qui allume la diode bicolore :

```
    BSF  LATB,4     ; la pression électrique sur RB4 sera forte
    BCF  LATB,5     ; la pression électrique sur RB5 sera faible

BOUCLE :
    GOTO   BOUCLE
```

La différence de pression électrique entre les broches RB4 et RB5 va induire un courant électrique de RB4 (haute pression) vers RB5 (basse pression). La diode va donc éclairer.

C. Utilisation du compilateur

- Démarrez le compilateur *Microchip MPLAB* (démarrer>>Microchip MPLAB IDE)
- Créez un nouveau projet :
  - ✓ Dans le menu *Project*, sélectionnez le champ *Project Wizard*
  - ✓ Sélectionnez le *PIC18F452*
  - ✓ Sous la rubrique *Active Toolsuite*, sélectionnez *Microchip MPASM Toolsuite et vérifiez les répertoires :*
    - o MPASM : C:\Program Files\MPLAB IDE\MCHIP_Tools\MPASMWIN.EXE
    - o MPLINK : C:\Program Files\MPLAB IDE\MCHIP_Tools\MPLINK.EXE
    - o MPLIB : C:\Program Files\MPLAB IDE\MCHIP_Tools\MPLIB.EXE
  - ✓ Nommez votre projet (par exemple theled) et sélectionnez votre répertoire de travail *R:\labo8*
  - ✓ Ajoutez au projet les fichiers « led.asm, firmware_v1.0 et picobit.lkr»
- Compilez votre projet (raccourci clavier F10)

D. Utilisation de l'environnement de programmation

Chaque robot possède un numéro d'identification. La station de base possède le numéro 0. Elle communique avec les robots par IR. De plus, tous les PC sont connectés à la station de base par le réseau ethernet. Ainsi, chaque utilisateur peut communiquer directement avec son robot depuis son PC.

- Ouvrez un shell dans Cygwin et placez-vous dans le répertoire
  `/home/david/gambit/gambc40b11/gsi`
- Entrez la commande (pour le robot 17) :
  `./robot.bat 17 /cygdrive/r/labo8/flash.hex`

Votre diode bicolore clignote ? C'est parfait, le plus dur est fait. Vous pouvez aborder les exercices. Sinon, faites appel aux démonstrateurs.

E. Exercices

<u>Mise en garde</u> : ne jamais faire d'accès aux registres PORTx, LATx ou TRISx autres que ceux qui commandent la diode bicolore. Vous risqueriez de créer des courts-circuits !!!

Utilisation de la diode bicolore.

La diode bicolore est très pratique pour indiquer ce qui se passe à l'intérieur du processeur. Essayez donc de voir comment elle réagit si vous lui appliquez d'autres configurations de pression électrique (FAIBLE-FAIBLE, FAIBLE-FORT, FORT-FAIBLE et FORT-FORT). N'oubliez pas de recompiler votre projet à chaque fois et d'envoyer ensuite le fichier à votre robot. Enfin, que se passe-t-il si vous changez la couleur de la diode des milliers de fois par seconde ? Toujours en utilisant le temps, pouvez-vous obtenir divers niveaux d'intensité lumineuse ?

Remarque : la commande de programmation est maintenant :

```
./robot.bat 17 /cygdrive/r/labo8/theled.hex
```

Utilisation de l'horloge temps réel

L'horloge interne du microcontrôleur équivaut à un compteur qui s'incrémente 100 fois par seconde. La précision est donc le 1/100 de seconde. La valeur du compteur est mémorisée sur 24 bits (3 octets). Ainsi, les valeurs :

00000000 00000000 00000001 = 1 centième de seconde
00000000 00000000 00000010 = 2 centièmes de seconde
00000000 00000000 00000011 = 3 centièmes de seconde
00000000 00000001 00000000 = 256 centièmes de seconde, soit 2.56 secondes
00000001 00000000 00000000 = 65536 centièmes de seconde, soit 655.36 secondes

Il faut aussi remarquer que le dernier bit (le bit 0) change à chaque centième de seconde. L'avant dernier bit (le bit 1) change tous les deux centièmes de secondes. Le bit 2 change tous les quatre centièmes de seconde … le bit i change tous les 2^i centièmes de seconde. La fonction qui permet de lire la valeur de l'horloge est :

```
call   fw_clock_read
```

Juste après son exécution, les 24 bits sont mémorisés dans les trois registres :

FW_VALUE_UP | FW_VALUE_HI | FW_VALUE_LO

Écrivez un programme qui fasse clignoter la diode du rouge au vert et réciproquement sur un cycle de 2.56 secondes (soit 1.28 seconde pour chaque couleur).

Envoi de messages au robot

Dès que vous appuyez sur une touche du clavier, un message avec le code ASCII de la touche est envoyé au robot. La commande qui vous permet de lire un message est :

```
call  fw_ir_rx_stdio_char
```

Si un nouveau message est arrivé, le bit de status C est activé. La valeur du caractère reçu est placée dans le registre WREG et une copie est aussi placée dans le registre FW_VALUE_LO. S'il n'y a pas de nouveau message, le bit de status C est désactivé. Pour attendre un nouveau message, on peut donc écrire :

```
tele_wait
  call  fw_ir_rx_stdio_char
  bnc tele_wait
```

Écrivez un programme qui permet de télécommander l'état de la diode :
'v' : la diode s'allume en vert
'r' : la diode s'allume en rouge
'e' : la diode s'éteint

Commande du robot

Le déplacement du robot est régi par la commande des vitesses de chacune de ses deux roues. Différentes vitesses sont possibles (en positif et en négatif). Lorsque les vitesses sont les mêmes, le robot avance (vitesses positives) ou recule (vitesses négatives) en ligne « presque » droite. Si les vitesses sont différentes, le robot tourne en conséquence. Si les vitesses sont carrément opposées, il tourne sur place. La commande qui permet de modifier la vitesse d'un moteur est :

```
  call  fw_motor
```

Les paramètres sont passés par les registres suivants :
MOTOR_ID = numéro du moteur (0 pour gauche, 2 pour droit)
MOTOR_ROT = sens de rotation , from (1 = avant, -1 = arrière, 0 = roue libre)
MOTOR_POW = puissance : (1-4 pour fournir de la puissance, -4 à -1 pour freiner)

Ainsi, pour faire avancer le robot à vitesse maximale, on peut exécuter le programme suivant :

```
  movlw 0
  movwf MOTOR_ID
  movlw 1
  movwf MOTOR_ROT
  movlw 4
  movwf MOTOR_POW
  call    fw_motor

  movlw 2
  movwf MOTOR_ID
  movlw 1
  movwf MOTOR_ROT
  movlw 4
  movwf MOTOR_POW
  call    fw_motor
```

Écrivez un programme permettant de télécommander le robot :
'g' : le robot tourne à gauche
'd' : le robot tourne à droite
'a' : le robot avance

‘r’ : le robot recule
‘s’ : le robot stoppe

Dans un premier temps en tous cas, utilisez une faible vitesse pour éviter des collisions avec d’autres robots. Souvenez-vous que les IR ne sont actifs que sur plate-forme. Il est donc impossible de sortir les robots de celle-ci. Par ailleurs, la bande passante étant assez faible, vous ne pourrez pas envoyez plus de deux commandes à la seconde dans les meilleurs cas.

Capteur de luminosité et envoi de messages

La lecture de la luminosité se fait par la commande :

```
call    fw_light_read
```

Le résultat de la lecture est placé dans WREG et une copie est placée dans le registre FW_VALUE_LO. L’envoi d’un message vers la station de base se fait par la commande :

```
call    fw_ir_tx_stdio_char
```

Le caractère à envoyer doit être place dans le registre WREG.

Ainsi, le programme suivant permet de lire la luminosité et de l’envoyer à la console :

```
call    fw_light_read
call    fw_ir_tx_stdio_char
```

Ajoutez une commande ‘l’ au programme de commande du robot qui vous permette de mesurer la luminosité et utilisez la pour rechercher manuellement la luminosité maximale (en direction de la lampe placée sur plateforme).