RAPPORT DE TP

TP 3 : Convertisseur A/D

Auteurs :

Baptiste BOUVAT
Gabriel GOMEZ
Thibaud CLÉMENT-LACROIX

Enseignant :

M. MOKHBER

Task 3.1

En annexe figure 1 le code en assembleur qui permet de convertir l'entrée RA0 en 8 bits. On doit en premier configurer le port A comme entrée analogique. Après on configure ADCON1 avec l'octet '*00001110*' et ADCON0 on met tous les bits à 0. Après on doit configurer les les interruptions du convertisseur ADIE, PEIE et GIE à 1 avec '*bsf*'. Pour lire les valeurs après on doit mettre à 1 le bit GO_DONE de ADCON0 et attribuer au port C le registre ADRESH à chaque interruption.

Task 3.2

Pour changer la tension de référence à 3,3V on doit faire '*bsf*' au bit VCFG0 de ADCON1.

Maintenant notre conversion est faite avec une plage plus petite de tension, 3,3V au lieu de 5V, donc elle sera plus précise.

On obtient donc le code en annexe figure 2 le code modifié avec une tension de référence 3,3V

Question 3.1

Effectivement le convertisseur du PIC18F4550 donne un résultat en 10 bits alors que le PIC marche sur 8 bits. Afin de pouvoir utiliser correctement les 10 bits le PIC va enregistrer le résultat en deux registres de 8 bits, donc va stocker les 10 bits de résultats dans une variable de 16 bits. On peut uniquement utiliser 8 bits en utilisant uniquement l'octet de poids fort (MSB) '*ADRESH*' ou l'octet de poids faible (LSB) '*ADRESL*'. Pour le cas du convertisseur ce sera plus pertinent de prendre l'octet de poids fort car même si cela est moins précis que prendre les 10 bits en entier le résultat sera proportionnel à la tension. Le cas contraire pour l'octet de poids faible, car le résultat ne reflète pas la vrai tension.

Pour savoir si c'est plus pertinent d'utiliser une tension de référence de 5V ou de 3,3V on doit faire des calculs:

Sur 8 bits et avec 5V on a :

$$\frac{5}{256} = 19,53\ mV$$

Sur 8 bits avec 3V on a :

$\frac{3,3}{256} = 12,89\ mV$

Lorsqu'on utilise 3,3V comme tension de référence le plus petit changement détectable est de 12,89 mV contre 19,53 mV si on utilise 5V. Donc le fait d' utiliser une tension de référence plus basse rend la conversion plus précise.

Question 3.2

PCFG3:0 sont des bits de configuration des ports A/D situé dans le registre ADCON1. Il permet de définir quels pin sont utilisés en entrée analogique.

`PCFG3:0 = 0000` → tous les AN sont analogiques

`PCFG3:0 = 1111` → tous les AN sont numérique

CHS3:0 est le bit déterminant quel canal analogique le pic 18f4550 va être utilisé entre AN0 et AN15.

`CHS3:0 = 0000` → sélectionne AN0
`CHS3:0 = 0101` → sélectionne AN5

Task 3.3

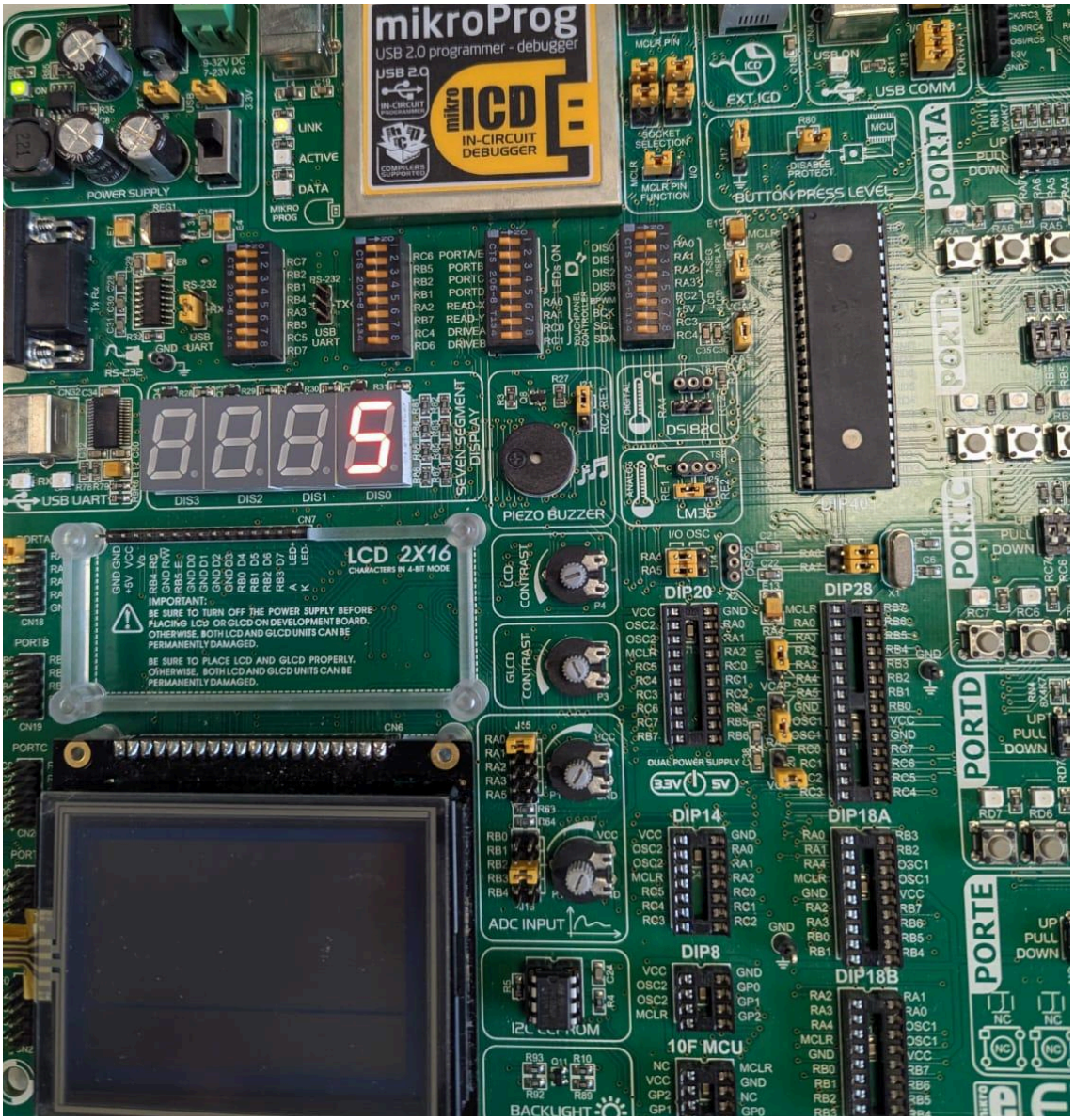
L'affichage sur les 7-segment se fait grâce à deux port le PORTD qui sert à sélectionner le segment ou les segments voulu et le PORTA qui permet de sélectionner le digit entre DIS0 et DIS3.

Pour afficher 5 ont utilise la table suivante

7-SEG DISPLAY OUTPUT	BINARY INPUT				DECODER OUTPUT						
	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1

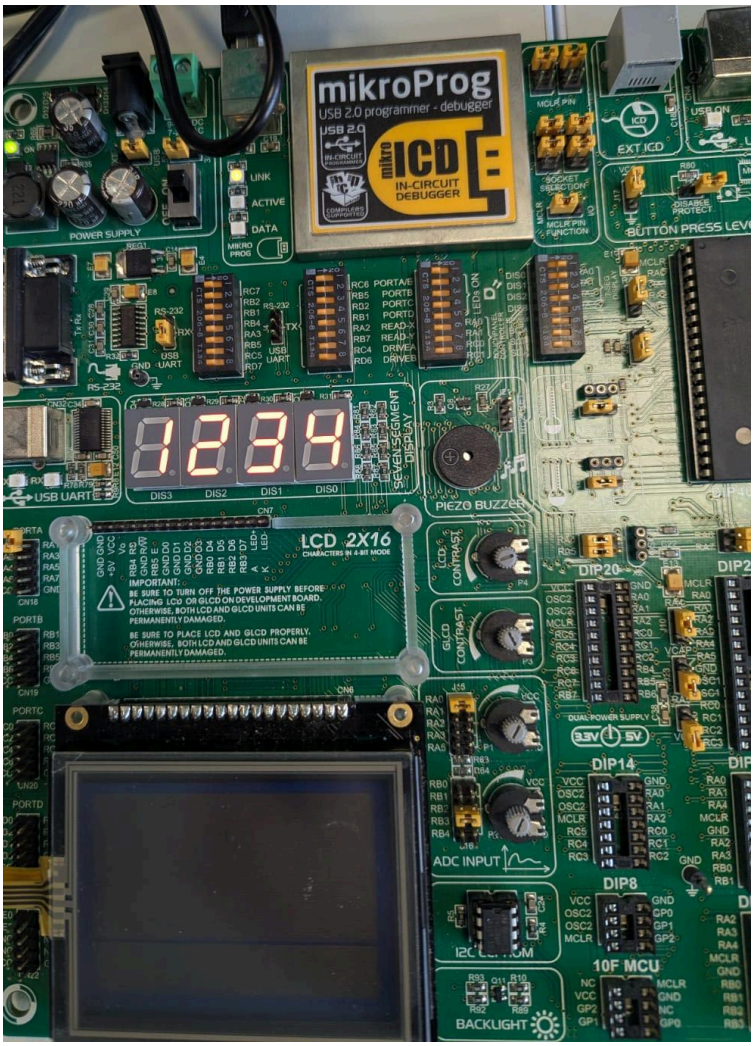
qui nous permet de savoir pour chaque digit quel code envoyer au PORTD pour allumer les bon segments.

on obtient donc le code en annexe figure 3.



Task 3.4

Pour afficher 4 digits différents sur les 7-segment il faut les parcourir un à un. En effet on ne dispose que d'un seul port pour envoyer des informations au digit, ils sont tous reliés ensemble. Pour simuler leur affichage simultanément on les parcourt très rapidement en affichant le digit souhaiter de façon à ce que l'œil humain les voient tous allumer. On propose donc le code en annexe figure 4 .



Task 3.5

Le programme fourni permettait uniquement d'afficher les secondes de 0 à 9 sur un afficheur 7 segments à l'aide d'un compteur unique (counter0) incrémenté à chaque interruption du Timer0. Pour réaliser le format d'horlge demandé (MM:SS) nous avons ajouté plusieurs compteurs (counter1, counter2, counter3) et modifié la routine d'interruption afin de gérer la progression des secondes et des minutes au format MM:SS. Concrètement, lorsque counter0 atteint 9, il est remis à zéro et counter1 est incrémenté (dizaines de secondes) ; quand counter1 atteint 5, il est à son tour réinitialisé et counter2 (unités de minutes) augmente, puis counter3 (dizaines de minutes) après 9 minutes. Ces changements permettent donc de faire défiler correctement les secondes et minutes sur les quatre afficheurs 7 segments :

```
TMR_INTERRUPT
    bcf INTCON, TMR0IF
    incf counter0, f
    movlw .9
    cpfsgt counter0
    goto skip_inc
    clrf counter0
    incf counter1, f
    movlw .5
    cpfsgt counter1
    goto skip_inc
    clrf counter1
    incf counter2, f
    movlw .9
    cpfsgt counter2
    goto skip_inc
    clrf counter2
    incf counter3, f
```

Annexes

Documents volumineux, éventuels codes (pas de code dans le rapport).

```
#include <p18F4550.inc>
CONFIG WDT=OFF ; disable watchdog timer
CONFIG MCLRE = ON ; MCLEAR Pin on
CONFIG DEBUG = OFF ; Disable Debug Mode
CONFIG FOSC = HS ; oscillator mode
org 0x0000 ; reset vector
goto INIT
org 0x0008 ; interrupt vector
goto irq_handle

irq_handle
    btfsc PIR1, ADIF ; is it AD?
    goto AD_interrupt ; yes
    retfie ; no, return f.i.

AD_interrupt
    bcf PIR1, ADIF
    movff ADRESH, PORTC
    retfie

INIT
    movlw   0xFF
    movwf   TRISA
    ; --- DDDDDDDA - only RA0 analog
    movlw b'00001110'
    movwf ADCON1
    ; --- enable AD interrupt
    bsf PIE1, ADIE
    bsf INTCON, PEIE
    bsf INTCON, GIE
    movlw b'00000000'
    movwf ADCON0
    bsf ADCON0, ADON
    clrf TRISC ; PORTC is an output
    clrf PORTC ; clear PORTC
    GOTO MAIN_LOOP

MAIN_LOOP
    bsf ADCON0, GO_DONE
    GOTO MAIN_LOOP
END
```

Figure 1: Code convertisseur RA0 sur 8 bits

```
#include <p18F4550.inc>
CONFIG WDT=OFF ; disable watchdog timer
CONFIG MCLRE = ON ; MCLEAR Pin on
CONFIG DEBUG = OFF ; Disable Debug Mode
CONFIG FOSC = HS ; oscillator mode
org 0x0000 ; reset vector
goto INIT
org 0x0008 ; interrupt vector
goto irq_handle

irq_handle
    btfsc PIR1, ADIF ; is it AD?
    goto AD_interrupt ; yes
    retfie ; no, return f.i.

AD_interrupt
    bcf PIR1, ADIF
    movff ADRESH, PORTD
    retfie

INIT
    movlw   0xFF
    movwf   TRISA
    ; --- DDDDDDDA - only RA0 analog
    movlw b'00001110'
    movwf ADCON1
    ; --- enable AD interrupt
    bsf PIE1, ADIE
    bsf INTCON, PEIE
    bsf INTCON, GIE
    movlw b'00000000'
    movwf ADCON0
    bsf ADCON0, ADON
    clrf TRISD ; PORTC is an output
    clrf PORTD ; clear PORTC
    bsf ADCON1, VCFG0
    GOTO MAIN_LOOP

MAIN_LOOP
    bsf ADCON0, GO_DONE
    GOTO MAIN_LOOP
END
```

Figure 2: Code convertisseur RA0 sur 8 bits (3,3V)

```
#include <p18F4550.inc>
CONFIG WDT=OFF ; disable watchdog timer
CONFIG MCLRE = ON ; MCLEAR Pin on
CONFIG DEBUG = OFF ; Disable Debug Mode
CONFIG FOSC = HS ; oscillator mode
org 0x0000 ; reset vector
goto INIT

INIT
    clrf TRISD
    clrf PORTD
    movlw b'1101101'
    movwf PORTD


MAIN_LOOP
    GOTO MAIN_LOOP
END
```

Figure 3: Code affichage du digit 5

```
MAIN_LOOP
    movlw b'0000110'
    movwf PORTD
    bsf PORTA, RA3
    CALL DELAI
    clrf PORTA
    movlw b'1011011'
    movwf PORTD
    bsf PORTA, RA2
    CALL DELAI
    clrf PORTA
    movlw b'1001111'
    movwf PORTD
    bsf PORTA, RA1
    CALL DELAI
    clrf PORTA
    movlw b'1100110'
    movwf PORTD
    bsf PORTA, RA0
    CALL DELAI
    clrf PORTA
    GOTO MAIN_LOOP

DELAI
    MOVLW   D'100'
    MOVWF   TEMP

D1
    DECFSZ  TEMP, F
    GOTO    D1
    RETURN
END
```

Figure 3: Code affichage du des digit 1 2 3 4