# Travaux Pratiques Simulation de programmes en C sur PIC 18F
# TP n°1

Ne pas oublier
Project->Build Options -> Project -> Onglet Directories
Choisir Library Search Path et le positionner à MCC18\lib

## I) Introduction

Soit le programme suivant :

```
#include <p18f452.h>
#pragma config WDT=OFF
void main(void){
      TRISA=0xFF; //PORTA en entrée
      TRISB=0x00; //PORTB en sortie
      while(1)
      {
            if (PORTA & 0x10) PORTB=1;
            else PORTB=0;
      }
}
```

Debugger -> Stimulus -> New Workbook puis l'onglet asynch (asynchrone) mettre RA4 en PIN/SFR avec l'action Toggle

View -> Watch ajouter PORTA et PORTB en SFR

Debugger -> Settings -> onglet « Animation/ Realtime Updates » cocher « Enable Realtime watch updates »

## II) Le printf et le scanf sur microcontroleur

### 1°) Le printf

La sortie d'un printf se fait sur la liaison série. Pour la voir dans le simulateur il faut configurer :

Debugger -> Settings -> onglet « UART1 IO » cocher « Enable UART1 IO »

Esssayer

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
void main(void){
      printf("Bonjour\n");
      while(1);

}
```

### 2°) Comment remplacer le scanf

On crée un fichier texte "demo.txt" qui contient :

```
wait 20 ms
"0123"
wait 10 ms
"0456"
```

Debugger -> Settings -> onglet « UART1 IO » après avoir coché « Enable UART1 IO »  "browse" pour trouver le fichier "demo.txt"

```
// Exercice 3 du poly TD
#include <p18f452.h>
#include <stdio.h>
#include <usart.h>
#include <stdlib.h>
#pragma config WDT = OFF
void main(void){
      int x,y;
      char chaine1[15],chaine2[15];
      // configure USART
      SPBRG = 25; // configure la vitesse (BAUD) 9600 N 8 1
      TXSTA = 0x24;
      RCSTA = 0x90;     // active l'USART
      printf("Bonjour\n");
      printf("Valeur de x ? :\n ");
      //scanf("%d",&x); en c windows ou Linux
      getsUSART(chaine1,4);
      chaine1[4]=0;
      x=atoi(chaine1);
      printf("Valeur de y ? :\n ");
      //scanf("%d",&y);
      getsUSART(chaine2,4);
      chaine2[4]=0;
      y=atoi(chaine2);
      printf("la somme de %d + %d = %d \n",x,y,x+y);
      CloseUSART();
      while(1);
}
```

Faire fonctionner ce programme avec plusieurs valeurs d'entrées pour vous approprier correctement de l'environnement.
Essayer les trois autres opérations : -, * et /.
Essayer ensuite la post-incrémentation et la pré-incrémentation (exercice 2 du poly)
Essayer aussi les opérateurs de décalages à gauche comme à droite:

```
...
void main(void){
      int x,y;
      x=1;
      y=x<<2;
      printf("Bonjour\n");
      printf("Valeur de y = %d\n",y);
      ....
      while(1);
}
```

Les masques seront expliqués pour réaliser l'exercice 5 p 4 du poly de Tds.
**Remarque** : l'initialisation de l'UART peut être faite avec :

```
OpenUSART( USART_TX_INT_OFF  |
           USART_RX_INT_OFF  |
           USART_ASYNCH_MODE |
           USART_EIGHT_BIT   |
           USART_CONT_RX     |
           USART_BRGH_HIGH,25 );
```

# Travaux Pratiques Simulation de programmes en C sur PIC 18F
# TP n°2

## Tableaux de données

Soit le programme suivant qui remplit un tableau avec des valeurs venant de l'extérieur :

```
#include <p18f452.h>
#include <stdio.h>
#include <usart.h>
#include <stdlib.h>
#pragma config WDT=OFF
void main(void) {
      int tab[20],i;
      char chaine[15];
      OpenUSART( USART_TX_INT_OFF  |
             USART_RX_INT_OFF  |
             USART_ASYNCH_MODE |
             USART_EIGHT_BIT   |
             USART_CONT_RX     |
             USART_BRGH_HIGH,25 );
      printf("Bonjour\n");
      i=0;
      do {
            printf("Valeur de x ? :\n ");
      //scanf("%d",&x); en c windows ou Linux
            getsUSART(chaine,4);
            chaine[4]=0;
            tab[i]=atoi(chaine);
            i++;
      }while(tab[i-1]!=0);
      CloseUSART();
      while(1);
}
```

Sa boucle se termine lorsque la valeur entrée est 0 et ce 0 ne fait pas partie des valeurs entrées.
On pourra donc lui donner un fichier texte en entrée de UART1 IO du genre :

```
wait 10 ms
"0123"
wait 10 ms
"0432"
wait 10 ms
"0433"
wait 10 ms
"0434"
wait 10 ms
"0000"
```

1°) Modifier le programme pour qu'il calcule la somme des valeurs entrées. Il sera naturellement intéressant d'afficher le résultat correspondant.
2°) Modifier ce programme pour qu'il calcule la moyenne des valeurs entrées. Ce sera un nombre à virgule donc un type « float » qui s'affiche avec un `"%f"`.
3°) Modifier ce programme pour qu'il calcule les extréma (maximum et minimum)
4°) Modifier ce programme pour qu'il calcule l'écart type : $\sigma = \sqrt{\frac{1}{N}\sum_{i=0}^{N} X_i^2 - \left(\frac{1}{N}\sum_{i=0}^{N} X_i\right)^2}$ et l'affiche.

# Travaux Pratiques Simulation de programmes en C sur PIC 18F
# TP n°3

## Tableaux de données (Correction partielle du TP2)

```
#include <p18f452.h>
#include <stdio.h>
#include <usart.h>
#include <stdlib.h>
#pragma config WDT=OFF
void main(void) {
  int tab[20],i,somme,j;
  float moyenne;
  char chaine[15];
// a compléter avec exemple TP2
  do {
// a compléter avec exemple TP2

   }while(tab[i-1]!=0);
      CloseUSART();
      somme=0;
      for(j=0;j<i-1;j++) {
            somme=somme+tab[j];
      }
       moyenne=  somme / i-1;
      printf("La somme est %d\n",somme);
      //printf("La moyenne est %f\n",(moyenne));
// demander à l'enseignant pour afficher la moyenne
      while(1);
}
```

## II) Chaînes de caractères

Pour entrer une chaîne de caractères on peut procéder comme précédemment avec l'USART1 mais cette méthode a l'inconvénient de nécessiter une chaine de longueur constante. On peut déclarer un tableau avec initialisation. L'affichage se fait avec un "%s" dans un printf. La longueur d'une chaine s'obtient avec strlen.

1°) Écrire un programme qui inverse une chaîne de caractères. (Attention au 0 en fin de chaîne)

```
#include <p18f452.h>
#include <stdio.h>
#include <string.h>
#pragma config WDT=OFF
void main(void) {
// declaration de la chaine initiale et resultat
  char resultat[25],chaine[]="Bonjour a tous";
// quelques variables intermédiaires
      char i,len;
// On trouve la longueur de chaine
      len=strlen(chaine);
// inverser la chaine ici
// avec une bonne boucle for
// puis on affiche le resultat
      printf("La chaine inversee est : %s\n",resultat);
      while(1);
}
```

2°) Modifier ce programme pour qu'il passe en majuscule toutes les lettres d'une phrase, puis toutes les premières lettres de mots seulement.

**Indications** : le code ascii de 'a' est 97 (0x61), celui de 'A' est 65 (0x41), celui de '0' est 48 (0x30). Le zéro de fin de chaîne est vraiment un zéro.

**Correction partielle** :

```
#include <p18f452.h>
#include <stdio.h>
#include <string.h>
#pragma config WDT=OFF
void main(void) {
// declaration de la chaine initiale et resultat
  char resultat[25],chaine[]="Bonjour a tous";
// quelques variables intermédiaires
      char i,len;
// On trouve la longueur de chaine
      len=strlen(chaine);
// inverser la chaine ici avec une bonne boucle for
      for(i=0;i<len;i++)
            resultat[i]=chaine[len-i-1];
      resultat[len]=0;
// puis on affiche le resultat
      printf("La chaine inversee est : %s\n",resultat);
      while(1);
}
```

# Travaux Pratiques Simulation de programmes en C sur PIC 18F
# TP n°4

## I) Introduction aux fonctions

On désire réaliser une fonction qui double une valeur donnée.

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
// declaration du prototype avant le main
float monDouble(float nb);
void main(void) {
      float result;
      result=monDouble(1.5);
      while(1);
}
// le code apres le main
float monDouble(float nb) {
  float local_nb;
  local_nb=nb*2;
  return local_nb;
}
```

**Remarques** : - le code de la fonction peut très bien être réalisé dans un fichier séparé.
- on exécute puis arrête. Le placement de la souris sur "result" permet de voir sa valeur.

## II) Calcul d'une racine carrée

On désire mettre en œuvre l'algorithme récursif de Héron d'Alexandrie (premier siècle de notre ère) pour le calcul d'une racine carrée. La récursion peut s'écrire simplement : $R_n=\frac{1}{2}(R_{n-1}+\frac{a}{R_{n-1}})$ où a est le nombre dont on cherche la racine.

1°) Écrire un programme sans fonction qui réalise le calcul comme ci-dessous.

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
void main(void) {
      float result,nb,erreur,precision;
      result=1;
      nb=2;
      precision=0.0001;
      do {
            result=0.5*(result+nb/result);
            erreur = result*result-nb;
            if (erreur < 0) erreur = -erreur;
      }while (erreur>precision);
      while(1);
}
```

2°) Modifier ce programme pour que la valeur nb soit demandée par l'intermédiaire de USART1. Et qu'il affiche le nombre de boucles réalisé.

```
wait 20 ms
"05.5"
```

```
// scanf("%f",&nb); n'existe pas et doit être remplace
par
getsUSART(chaine,4);//nb sur 4 caracteres
chaine[4]=0;
nb=atof(chaine); // prototype dans stdlib.h
```

3°) Modifier ce programme pour qu'il utilise une fonction de prototype "float racine(float nb);"

4°) Écrire une fonction factorielle ainsi qu'un programme qui l'utilise.

**Correction partielle** :

```
#include <p18f452.h>
#include <stdio.h>
#include <stdlib.h> //pour atof
#include <usart.h> // pour getsUSART
#pragma config WDT=OFF
void main(void) {
      float result,nb,erreur,precision;
      char chaine[10];
// configure USART
      SPBRG = 25; // configure la vitesse (BAUD) 9600 N 8 1
      TXSTA = 0x24;
      RCSTA = 0x90;     // active l'USART
      result=1;
      getsUSART(chaine,4);//nb sur 4 caracteres
      chaine[4]=0;
      nb=atof(chaine);
      precision=0.0001;
      do {
            result=0.5*(result+nb/result);
            erreur = result*result-nb;
            if (erreur < 0) erreur = -erreur;
      }while (erreur>precision);
      while(1);
}
```

# Travaux Pratiques Simulation de programmes en C sur PIC 18F
# TP n°5 (deux séances)

## I) Utilisation d'un sous-programme

Soit le sous-programme donné ci-après dont l'objectif est d'afficher le contenu d'un registre :

```
void affiche(char port){
  char i;
  for (i=7;i>=0;i--)
    if ((port & (1<<i))== (1<<i))
     printf("*|");
    else
      printf("  |");
  printf("\n");
}
```

L'affichage, comme d'habitude, se fait par la sortie série dans "Output" dans l'onglet "SIM Uart1".
Réaliser un programme principal qui réalise un compteur binaire et l'affiche en complétant le programme ci-dessous.

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
// declaration du prototype avant le main
void affiche(char port);

void main(void) {
// configure USART : autre methode
      SPBRG = 25; // configure la vitesse (BAUD) 9600 N 8 1
      TXSTA = 0x24;
      RCSTA = 0x90;     // active l'USART
// A completer ici
// .....
    while(1);
}
```

A partir de maintenant on suppose qu'une variable char sera calculée puis sortie sur un port lui-même relié à des LEDs. C'est pour cela que l'on parlera de chenillars (LEDs allumée se déplaçant).

## II) Chenillar simple

Un chenillar simple est réalisé avec un programme principal contenant :

```
char port; // déclaration

for (i=0;i<8;i++){
      port = 1 << i;
      affiche(port);
    // attente a adapter a votre processeur
      for (j=0;j<20000000;j++);
    }
```

La boucle d'attente n'est pas nécessaire pour notre simulation car on a une sortie mise à jour en temps réel, certes, mais que l'on aura tout loisir de lire après coup.

## III) Chenillars complexes

1°) Modifier le programme pour qu'il fasse un chenillar double avec croisement.
2°) Modifier ce programme pour qu'il fasse un chenillar à entassement.

Correction

Exo1

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
void affiche(char port);
void main(void) {
  char nb;
  SPBRG = 25;
  TXSTA = 0x24;
  RCSTA = 0x90;
  for (nb=0;nb<25;nb++)
     affiche(nb);
  while(1);
}
void affiche(char port){
  char i;
  for (i=7;i>=0;i--)
      if((port & (1<<i))==(1<<i))
            printf("*|");
      else
            printf(" |");
  printf("\n");
}
```

exo2

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
void affiche(char port);
void main(void) {
  char port,i;
  int j;
  SPBRG = 25;
  TXSTA = 0x24;
  RCSTA = 0x90;
  for (i=0;i<8;i++){
      port=1<<i;
    affiche(port);
      for (j=0;j<30000;j++);
  }
  while(1);
}
void affiche(char port){
  char i;
  for (i=7;i>=0;i--)
      if((port & (1<<i))==(1<<i))
            printf("*|");
      else
            printf(" |");
  printf("\n");
}
```

exo3

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
void affiche(char port);
void main(void) {
  char port,i;
```

```
  int j;
  SPBRG = 25;
  TXSTA = 0x24;
  RCSTA = 0x90;
  for (i=0;i<8;i++){
      port=1<<i|128>>i;
    affiche(port);
      for (j=0;j<30000;j++);
  }
  while(1);
}
void affiche(char port){
  char i;
  for (i=7;i>=0;i--)
      if((port & (1<<i))==(1<<i))
            printf("*|");
      else
            printf(" |");
  printf("\n");
}
```

exo4

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT=OFF
void affiche(char port);
void main(void) {
  char port,i,k;
  char tab[8]={0xFE,0xFC,0xF8,0xF0,0xE0,0xC0,0x80,0};
  int j;
  SPBRG = 25;
  TXSTA = 0x24;
  RCSTA = 0x90;
  for (i=7;i>-1;i--){
      for(k=0;k<i+1;k++){
            port=1<<k | tab[i];
      affiche(port);
            for (j=0;j<30000;j++);
      }
  }
  while(1);
}
void affiche(char port){
  char i;
  for (i=7;i>=0;i--)
      if((port & (1<<i))==(1<<i))
            printf("*|");
      else
            printf(" |");
  printf("\n");
}
```

# Travaux Pratiques Simulation de programmes en C sur PIC 18F
# TP n°6

## I) Utilisation d'un autre sous-programme pour affichage sur sept segments

Un afficheur sept segments est géré ("simulé") par le sous programme appelé

```
void affiche7segs(char port);
```

L'affichage, comme d'habitude, se fait par la sortie série dans "Output" dans l'onglet "SIM Uart1". Ce sous-programme est utilisé avec le programme principal ci-dessous destiné à afficher "bonjour".

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT = OFF
void affiche7segs(char port);
void main(void) {
      char i;
      char bonjour[7]={0x7C,0x3F,0x54,0xE,0x3F,0x1C,0x50};
      int j;
// configure USART
      SPBRG = 25; // configure la vitesse (BAUD) 9600 N 8 1
      TXSTA = 0x24;
      RCSTA = 0x90;     // active l'USART
      for (i=0;i<7;i++){
            affiche7segs(bonjour[i]);
            // attente a adapter a votre processeur
             for (j=0;j<30000;j++);
      }
      while(1);
}
```

Ce programme avec le sous-programme est disponible sur mon site internet :
http://pagesperso-orange.fr/moutou/tp6exo1.c

## II) Affichages simples

1°) Modifier le programme pour qu'il affiche les chiffres de 0 à 9.
2°) Modifier le programme pour qu'il affiche les chiffres de 0 à F.

## III) Modification du sous-programme d'affichage sept segments

3°) Modifier le programme pour qu'il permette l'affichage sur deux afficheurs commandés par un multiplexeur.

Le programme qui affiche "Bonjour"

```
#include <p18f452.h>
#include <stdio.h>
#pragma config WDT = OFF
void affiche7segs(char port);
void main(void) {
       char i;
       char bonjour[7]={0x7C,0x3F,0x54,0xE,0x3F,0x1C,0x50};
       int j;
// configure USART
       SPBRG = 25;    // configure la vitesse (BAUD) 9600 N 8 1
       TXSTA = 0x24;
       RCSTA = 0x90;  // active l'USART
       for (i=0;i<7;i++){
              affiche7segs(bonjour[i]);
              // attente a adapter a votre processeur
               for (j=0;j<30000;j++);
       }
       while(1);
}

void affiche7segs(char port){
printf((char *)"---------\n");
  if ((port & 1) == 1)
    printf(" **** \n");
  else
    printf("      \n");
  if ((port & 0x22)==0x22){
    printf("*    *\n");
    printf("*    *\n");
    printf("*    *\n");
  }
  if ((port & 0x22)==0x02){
    printf("     *\n");
    printf("     *\n");
    printf("     *\n");
  }
  if ((port & 0x22)==0x20){
    printf("*     \n");
    printf("*     \n");
    printf("*     \n");
  }
  if ((port & 0x22)==0x00){
    printf("      \n");
    printf("      \n");
    printf("      \n");
  }
  if ((port & 0x40)==0x40)
    printf(" **** \n");
  else
    printf("\n");
  if ((port & 0x14)==0x14){
    printf("*    *\n");
    printf("*    *\n");
    printf("*    *\n");
  }
  if ((port & 0x14)==0x04){
    printf("     *\n");
    printf("     *\n");
    printf("     *\n");
  }
  if ((port & 0x14)==0x10){
    printf("*     \n");
    printf("*     \n");
    printf("*     \n");
  }
  if ((port & 0x14)==0x00){
    printf("      \n");
    printf("      \n");
    printf("      \n");
  }
  if ((port & 0x08)==0x08)
    printf(" **** \n");
  else
    printf("\n");
printf((unsigned char *)"---------\n");
}
```