

Prise en main de MPLAB

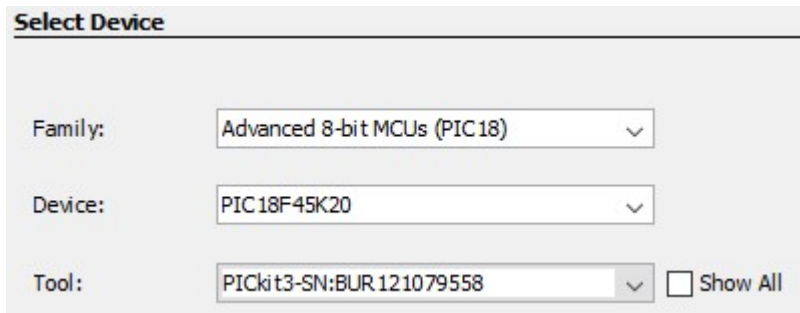
MPLAB X IDE doit être installé sur le poste

⇒ Lancer le logiciel

⇒ Brancher le PICkit3 et le PICkit Demo Board

Créer un nouveau projet

File ⇒ New Project ⇒ Microchip Embedded ⇒ Standalone Project



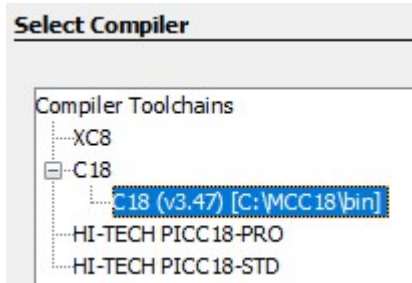
Select Device

Family: Advanced 8-bit MCUs (PIC18) ▾

Device: PIC18F45K20 ▾

Tool: PICkit3-SN: BUR121079558 ▾ Show All

⇒ Choisir le compilateur MCC18



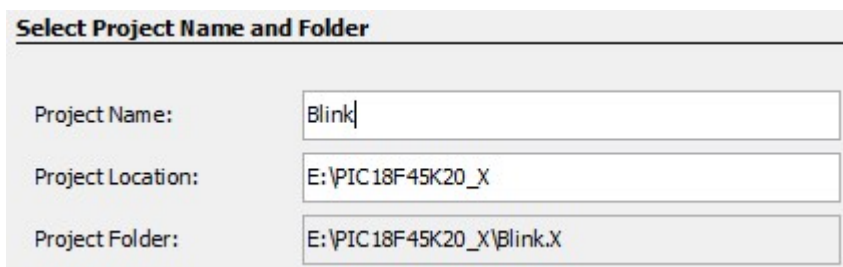
Select Compiler

Compiler Toolchains

- XC8
- C18
 - C18 (v3.47) [C:\MCC18\bin]**
 - HI-TECH PICC18-PRO
 - HI-TECH PICC18-STD

⇒ Créer un répertoire dans l'espace de travail : PIC18F45K20_X par exemple

⇒ Donner un nom au projet et le sauvegarder dans le répertoire créer



Select Project Name and Folder

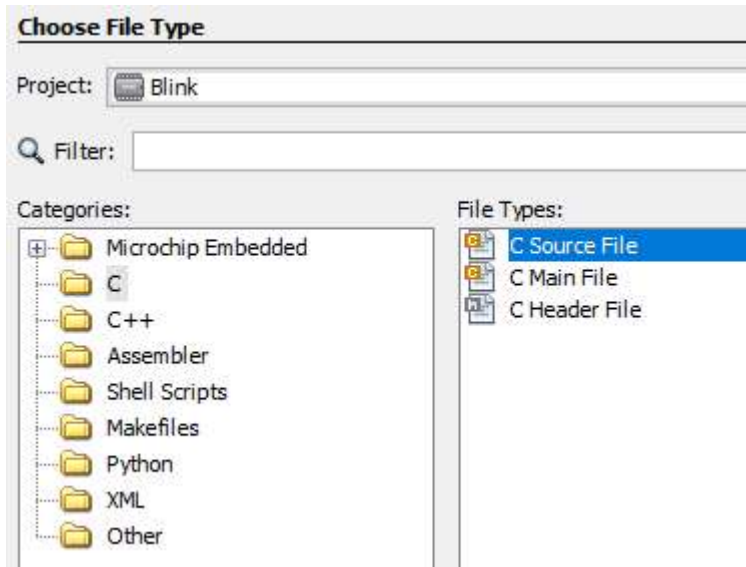
Project Name: Blink

Project Location: E:\PIC18F45K20_X

Project Folder: E:\PIC18F45K20_X\Blink.X

Ajouter des fichiers sources au projet

⇒ File ⇒ New ⇒ Choisir un fichier C



⇒ Donner le nom main (main.c avec l'extension).

⇒ Copier le code ci-dessous dans le fichier main.c

```
#include <stdio.h>
#include <stdlib.h>

#include <p18cxxx.h>
#include "HardwareProfile.h"
#include "tempo.h"

#define LED PORTDbits.RD0
#define LED_DIR TRISDbits.TRISD0

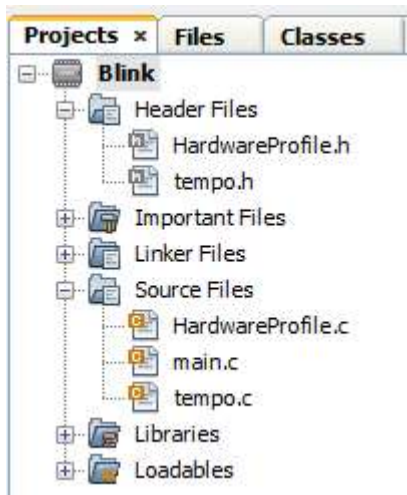
void main(void)
{
    init_Board();//initialisation par défaut
    LED_DIR=0;// déclaration du port en sortie

    while(1)
    {
        LED=0;
        tempo_ms(100);
        LED=1;
        tempo_ms(100);
    }
}
```

⇒ De la même manière, ajouter les fichiers HardwareProfile.c, HardwareProfile.h, tempo.c et tempo.h.

⇒ Copier le contenu de ces fichiers donnés en annexe.

Le projet doit alors être organisé de la manière suivante :



Compiler et exécuter le projet



⇒ Pour compiler le projet



⇒ Pour compiler et programmer le microcontrôleur

Il est possible d'alimenter le kit par l'intermédiaire du PICKIT3 : Window ⇒ Dashboard ⇒ PICKIT3 ⇒ Power ⇒ target circuit from PICKIT3 ⇒ 3V

Attention : cette manipulation n'est à faire que si la cible n'a pas d'autre alimentation

Project Properties - Blink

Categories:

- General
- File Inclusion/Exclusion
- Conf: [default]**
- PICKIT 3
- Loading
- Libraries
- Building
- C18 (Global Options)
- MPASMWIN
- mcc18
- MPLINK

Option categories: Power

Power target circuit from PICKIT3	<input checked="" type="checkbox"/>
Voltage Level	3.0

Débugger le projet



⇒ Permet de se mettre en mode Debug

Il est alors possible de mettre des points d'arrêt en cliquant sur la ligne de programme souhaitée

A chaque exécution, le programme s'arrête aux points d'arrêt et il est possible de le faire exécuter instruction par instruction (en rentrant ou pas dans les sous programmes).



Exécution du programme



Exécution instruction par instruction sans entrer dans les sous programmes



Exécution instruction par instruction en entrant dans les sous programmes

⇒ Placer 2 points d'arrêt comme ci-dessous et tester les différentes possibilités de débogage.

⇒ Faire une démo au prof. pour avoir des explications complémentaires.

```
15 |
16 |     while(1)
17 |     {
18 |         LED=0;
19 |         tempo_ms(100);
20 |         LED=1;
21 |         tempo_ms(100);
22 |     }
23 | }
```

Tester la commande en PWM

- ⇒ Créer un nouveau projet appelé pwm
- ⇒ Copier les 5 fichiers du projet précédent dans le nouveau répertoire pwm.x
- ⇒ Ajouter ces 5 fichiers au nouveau projet
- ⇒ Modifier le fichier main.c par le programme ci-dessous :

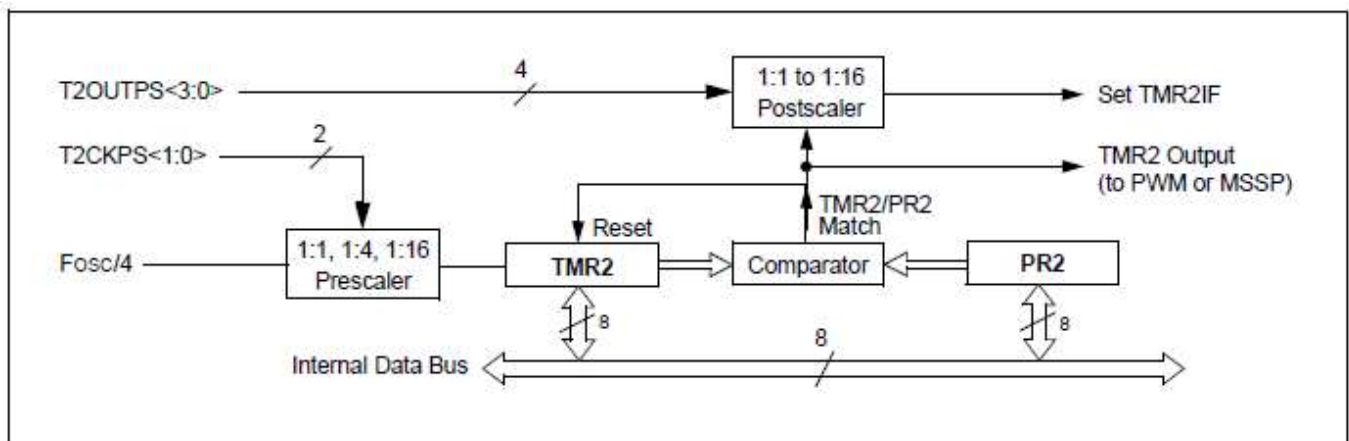
```
#include <stdio.h>
#include <stdlib.h>

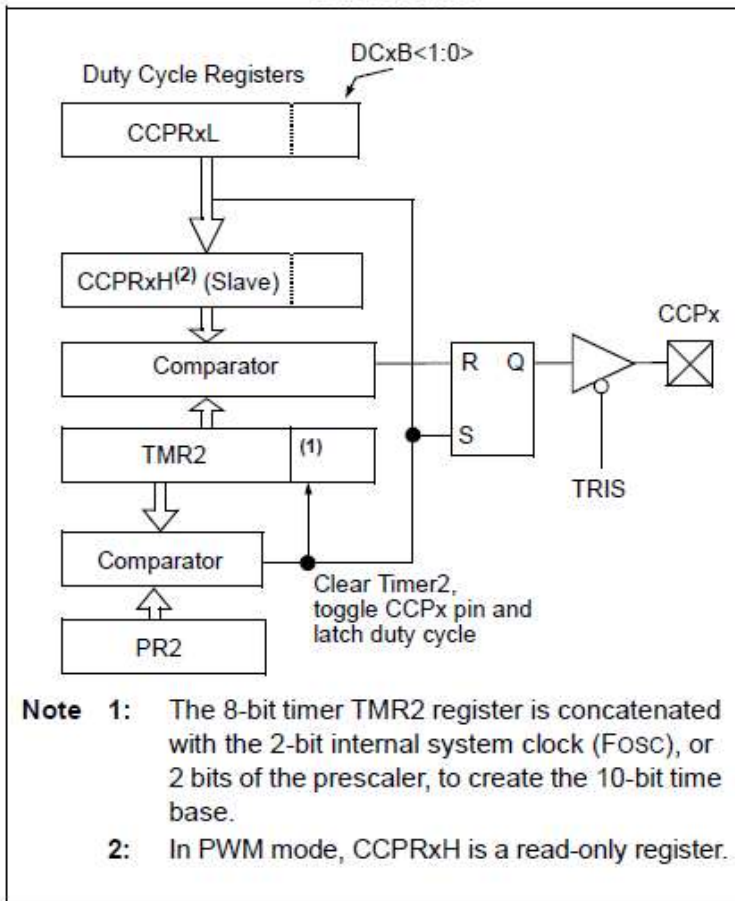
#include <p18cxxx.h>
#include "HardwareProfile.h"
#include "tempo.h"

void main(void)
{

    init_Board();//initialisation par défaut
    TRISCbits.TRISC2=0;
    PR2=125;
    CCPR1L=62;
    CCP1CON=0b00000000;
    T2CON=0b00000101;
    CCP1CON=0b00001100;
    while(1);
}
```

- ⇒ Relever à l'oscilloscope le signal sur RC2. Mesurer la fréquence et le rapport cyclique.
- ⇒ Justifier la valeur de ces mesures en sachant que F_{osc} est de 64MHz et en prenant en compte les structures suivantes (avec la valeur placée dans T2CON, le pré diviseur est programmé en 1:4)





⇒ Modifier le programme pour avoir une fréquence de 20kHz et un rapport cyclique de 25%. Tester.

HardwareProfile.c

```
#include <p18cxxx.h>
#include "HardwareProfile.h"

#if defined(__18F45K20)
    #pragma romdata CONFIG1H = 0x300001
    const rom unsigned char config1H = 0x08;

    #pragma romdata CONFIG2L = 0x300002
    const rom unsigned char config2L = 0x1F;
    #pragma romdata CONFIG2H = 0x300003
    const rom unsigned char config2H = 0x1E;

    #pragma romdata CONFIG3H = 0x300005
    const rom unsigned char config3H = 0x8B;

    #pragma romdata CONFIG4L = 0x300006
    const rom unsigned char config4L = 0x85;
#endif

void init_Board(void)
{

int i;

#if defined(__18F45K20)
    //validation PLL
    OSCCON=0b01110000;
    OSCTUNEbits.PLEN=1;
    for (i=0;i<1000;i++);
#endif

}
```

HardwareProfile.h

```
#define OSC_64MHZ
void init_Board(void);
```


tempo.h

```
void tempo_1us(void);  
void tempo_ms(int);  
void tempo_10us(void);  
void tempo_100us(void);
```