

Document de Cours et TP autorisés seulement autorisés

Les réponses ne doivent pas déborder les emplacements prévus -

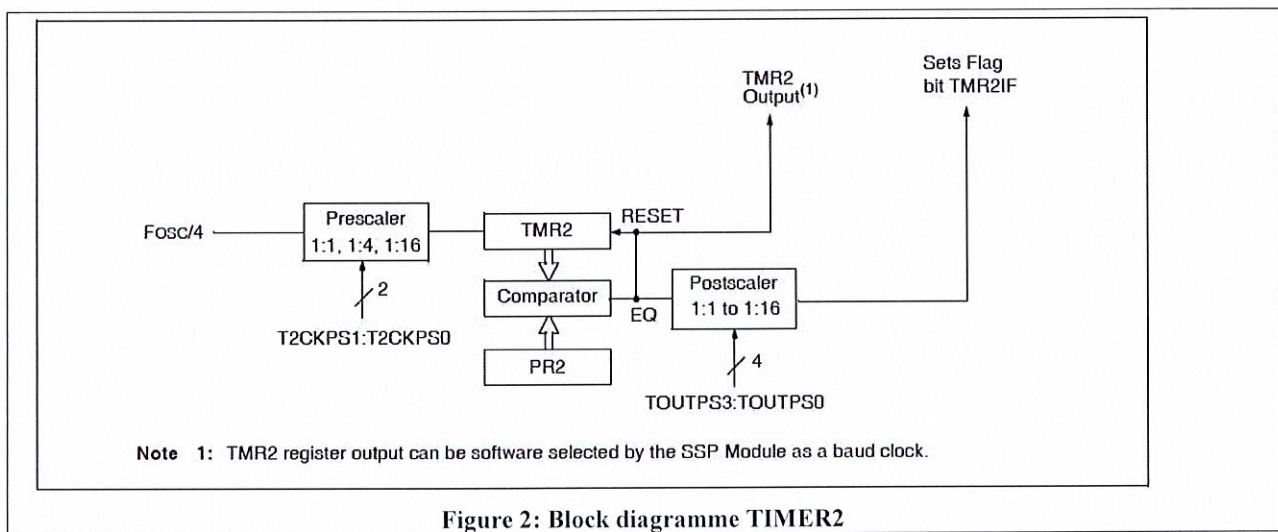
N° d'anonymat :

Exercice N° 1 (6 points)

Donner le programme équivalent en C en partant du principe que le programme en C est 10 fois plus lent que son équivalent en assembleur

<i>Programme Assembleur</i>	<i>Equivalent en C</i>
<pre> ***** ; ; LES Fichiers d'inclusion * ; #include <P18F4550.INC> ; fichier include ***** ; ; LES Directives config * ; CONFIG LVP = OFF ; CONFIG PBADEN = OFF ; port b en numérique CONFIG WDT = OFF ; Watch dog off CONFIG FOSC = HS ; HS oscillateur 20 Mhz ***** ; ; Déclaration des variables *** ; cblock 0x020 cmpt1: 1 ; Registre dédié à la BOUCLE1 cmpt2: 1 ; Registre dédié à la BOUCLE2 endc ***** ; ; DEMARRAGE SUR RESET ***** org 0x000 ; Adresse de départ après reset goto BEGIN ***** ; ; Sous-programme de temporisation de 500 µS TEMPO movlw d'250' movwf cmpt2 BOUCLE2 movlw d'10' movwf cmpt1 BOUCLE1 incfsz cmpt1,1 goto BOUCLE1 incfsz cmpt2,1 goto BOUCLE2 return ***** ; ; Ici, commence le programme *** call TEMPO ; Appel de la temporisation de 500 µS end </pre>	<pre> /**Merci de commenter vos instructions en C** </pre>

Exercice N° 2 (2 points)



1) Expliquer le fonctionnement du TIMER2

2) Donner la différence fondamentale avec le TIMER1

3) En fonctionnement en interruption expliquer le rôle du bit périphérique du registre INTCON

4) Expliquer le rôle et fonctionner des Prescaler et Postscaler :

- Prescaler:

Postscaler :

Exercice N° 3 (3 points)

1) Donner les différentes façons de programmer en C une interruption sur un 18F4550 en C, pour chaque cas donner un exemple

2) On écrit un programme en C dont la cible est le 18F4550. Donner la démarche suivie par le compilateur MPLAB pour créer le résultat final acceptable pour le PICDEM utilisé en TP

3) On vous donne une ligne de programme dont la cible est le 18f4550, expliquer la fonction et les paramètres d'entrée.

```
OpenTimer0(TIMER_INT_OFF & T0_8BIT & T0_SOURCE_INT & T0_PS_1_32);
```

Exercice N° 4 (5 points)

Un programme pour gérer un Timer a été mélangé par erreur. Le mélange se fait par bloc d'instruction. Le bloc commence par un titre compris entre `/*...*/`. Il faut récrire le programme dans le bon ordre pour qu'il puisse marcher correctement

Programme de Timer dans le désordre	Programme à Ordonner
<pre> //APPLICATION TESTE SUR TIMER EN UTILISANT LA BIBLIOTHEQUE /* Programme principal */ void main(void) { // Initialisation des variables // Validation interruption générale et périphérique INTCONbits.GIEH = 1; INTCONbits.PEIE = 1; // port utilisé TRISB=0; PORTB=0x00; //initialisation timer1 T1CON = 0b00000001; PIR1bits.TMR1IF = 0; PIE1bits.TMR1IE = 1; //variable test i=0; //Active le TIMER1 -- prescalerprescaler1:81:8 OpenTimer1(TIMER_INT_ON & T1_PS_1_1 & T1_16BIT_RW & T1_SOURCE_INT); WriteTimer1(65000); //initializer le timer0 while(1); } /* declaration variables */ int i; void WriteTimer1(unsigned int timer); void it_prioritaire(void); /* configuration du system */ #pragma config WDT=OFF // Watch dog off #pragma config PBADEN=OFF // Port B en digital IO #pragma config FOSC=HS // HS oscillator, #pragma config PWRT=ON // Power on timer #pragma config BOR=OFF // Brown out off #pragma config LVP=OFF // Pas de prog single supply #pragma config ICPRT=OFF // Port de debug #pragma config DEBUG=OFF // Debug off /* Routine Interruption */ #pragma code </pre>	

```

#pragma interrupt it_prioritaire
void it_prioritaire(void)

{
    while( ! PIR1bits.TMR1IF ); //Teste si le flag
    WriteTimer1(65000); //reinitialiser le timer0
    i=i+1; //verifier si le sous-programme est active
    // traitement interruption
    PORTBbits.RB2=! PORTBbits.RB2;
//complemente le bit numero 2
    //Annulation du FLAG
    PIR1bits.TMR1IF = 0;
}

/* header file */
#include <p18f4550.h>
#include <stdlib.h>
#include <stdio.h>
#include <timers.h>

/* Choix de type d interruption : 0x08 pour haute
priorite */
#pragma code address_it=0x08
void int_toto(void)
{
    _asm GOTO it_prioritaire _endasm
}

```

Exercice N° 5 (4 points)

1) Réaliser programme qui permet d'avoir une boucle infinie.

2) Réaliser de 2 façon un programme de temporisation en C :

- Méthode 1

- Méthode 2

3) Réaliser un programme qui comprends deux fonctions (fonct1 et fonct2) qui seront appelés dans un programme principal sous une condition (par exemple a=1, exécute fonct1 1, a=2, exécute fonct2)

4) Expliquer les particularités des deux fonctions ci-après :

int fonction_convertieur (int i, int coef) :

void main (void) :