Répondre directement sur les feuillets suivants. Ne pas oublier de mettre votre nom et de signer chaque feuillet.

Etude d'un programme implanté sur PIC 16F876

```
#define BOUTON PORTA.0
#define Out1 PORTA.1
#define Out2 PORTA.2

#include "int16CXX.H"

#pragma origin 4

interrupt int_server( void)
{
    int_save_registers
    if (T0IF)  {
        /* TMR0 overflow interrupt */
        TMR0 = 100;
        if (Out1 == 1)
            Out1 = 0;
        else
            Out1 = 1;
        T0IF = 0;  /* reset flag */
    }
    if (INTF)  {
        /* INT interrupt */
        INTF = 0;  /* reset flag */
    }
    if (RBIF)  {
        /* RB port change interrupt */
        W = PORTB; /* clear mismatch */
        RBIF = 0;  /* reset flag */
    }
    int_restore_registers
}

void main( void)
{
    ADCON1 = 0b0110; // PORTA digital
    PORTA = 0; //76543210
    TRISA =    0b11111001;
    OPTION = 1; /* prescaler division par 4 */
    TMR0 = 100;
    T0IE = 1;
    GIE = 1;    /* autorisation interruptions */

    while (BOUTON == 1);

    while (1)  {
        Out2 = 0;
        nop();
        nop();
        Out2 = 1;
    }
}
```

Nom : Prénom :	Signature :

Voici un programme souvent utilisé en exemple pour se familiariser à l'utilisation des interruptions. Ce programme est fonctionnel. Les documents utiles se trouvent en annexe. Les Mots Clé T0IF et T0IE utilisés dans le programme, correspondent à TMR0IE et TMR0IF de la documentation. Ils sont en relation avec le **compteur de 8 bits** TIMER0 qui s'incrémente automatiquement à partir de l'horloge du système (quartz) suivie d'une prédivision (PRESCALER). Le Microcontrôleur est un PIC 16F876.

Etude du programme

1) Où se trouve le programme principal ?

2) Comment sont réparties les entrées et sorties sur PORTA ?

3) Fonction interruption
Quels sont les sources d'interruption qui peuvent être prises en compte (mais pas forcément traitées) par la fonction d'interruption telle qu'elle est actuellement écrite?

4) En analysant juste la fonction interruption, quelle interruption est effectivement traitée par cette fonction ?

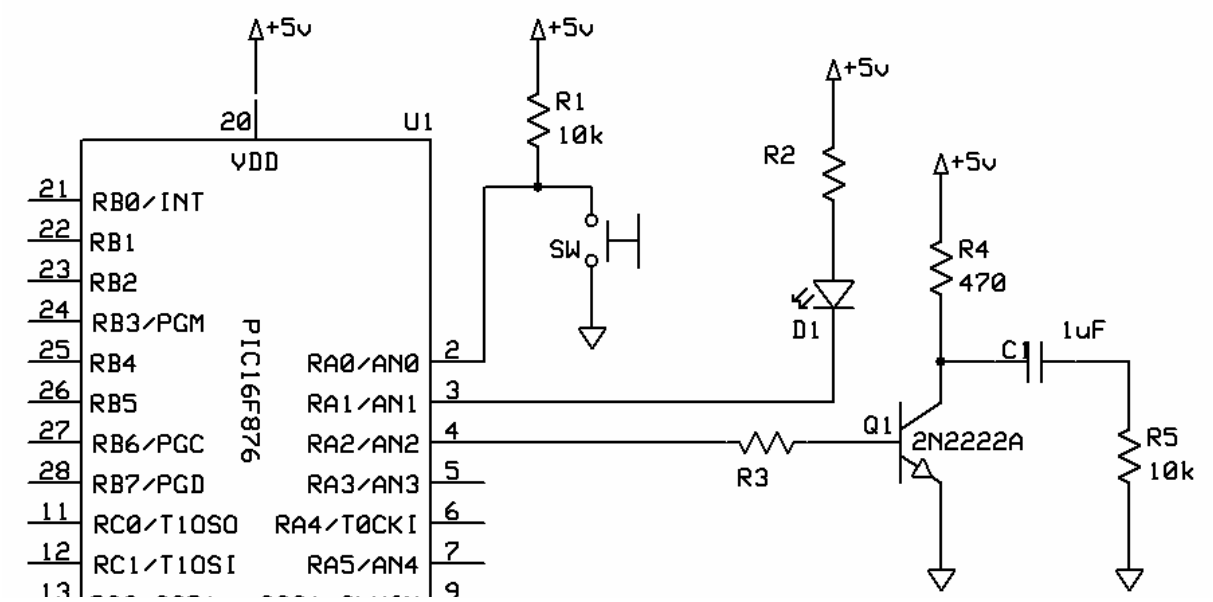
5) Quelle est l'utilité des macros prédéfinies `int_save_registers` et `int_restore_registers` ?

6) Comment est autorisée dans le programme principal l'interruption provoquée par le débordement du compteur TIMER0 ?

7) Tels que sont réglés les registres, le compteur TIMER0 s'incrémente à une fréquence de 250 kHz (le quartz d'horloge étant de 4 MHz). Le compteur TIMER0 étant chaque fois réinitialisé à une certaine valeur, quelle sera la fréquence des interruptions ?

Nom : Prénom :	Signature :

8) Quelle action est réalisée la partie de traitement de l'interruption ? Caractérisez la forme du signal sur PORTA.1.



9) Quel est l'état présent sur PORTA.0 lorsque le bouton SW est appuyé ?

10) Quelle est l'utilité de la résistance R1 dans le schéma ci-dessus ?

11) On désire faire circuler un courant de 5 mA dans la Led D1. La tension de seuil de cette LED étant de 1.4V, déterminez la valeur de R2.

Nom : Prénom :	Signature :

12) On souhaite que le transistor Q1, travaille en saturation. Quelle doit être la valeur maximale de R3 ?

13) Quelle est l'utilité du condensateur C1 ?

14) En supposant sur PORTA.2 (RA2) un signal périodique parfaitement carré, donnez son allure sur la broche 4 de U1 et aux bornes de R5

15) Que réalise la boucle "`while (BOUTON == 1);`" ?

16) Quelle est la particularité de la boucle " `while (1) { ` " ?

17) Que réalise la boucle suivante :

```
while (1)  {
    Out2 = 0;
    nop();
    nop();
    Out2 = 1;
}
```

Nom : Prénom :	Signature :

La boucle précédente ayant été compilée, on obtient les instructions assembleur suivantes :

``; while (1) {```	Nbre de cycles
``; Out2 = 0;```	
``m006 BCF 0x03,RP0```	→
``BCF 0x03,RP1```	→
``BCF PORTA,2```	→
``; nop();```	
``NOP```	→
``; nop();```	
``NOP```	→
``; Out2 = 1;```	
``BSF PORTA,2```	→
``; }```	
``GOTO m006```	→
	Total :

18) Quelle est la fréquence du signal généré sur la broche PORTA.2, sachant que le quartz de l'horloge est de 4 MHz, et qu'il faut 4 coups d'horloge pour faire un cycle machine (remplir le tableau ci-dessus) ?

19) Quel est le rapport cyclique du signal généré ? (Durée du signal à l'état haut divisé par la période)

20) Comment modifieriez-vous le programme en C pour avoir un rapport cyclique de 0.5 ?

Annexes

Jeu d'instructions du 16F87x

PIC16F87XA

TABLE 15-2: PIC16F87XA INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb			LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	f, d	Add W and f	1	00	0111	dfff	ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00	0101	dfff	ffff	Z	1,2
CLRF	f	Clear f	1	00	0001	lfff	ffff	Z	2
CLRW	-	Clear W	1	00	0001	0xxx	xxxx	Z	
COMF	f, d	Complement f	1	00	1001	dfff	ffff	Z	1,2
DECF	f, d	Decrement f	1	00	0011	dfff	ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00	1011	dfff	ffff		1,2,3
INCF	f, d	Increment f	1	00	1010	dfff	ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00	1111	dfff	ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00	0100	dfff	ffff	Z	1,2
MOVF	f, d	Move f	1	00	1000	dfff	ffff	Z	1,2
MOVWF	f	Move W to f	1	00	0000	lfff	ffff		
NOP	-	No Operation	1	00	0000	0xx0	0000		
RLF	f, d	Rotate Left f through Carry	1	00	1101	dfff	ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00	1100	dfff	ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00	0010	dfff	ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00	1110	dfff	ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00	0110	dfff	ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS									
BCF	f, b	Bit Clear f	1	01	00bb	bfff	ffff		1,2
BSF	f, b	Bit Set f	1	01	01bb	bfff	ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01	10bb	bfff	ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01	11bb	bfff	ffff		3
LITERAL AND CONTROL OPERATIONS									
ADDLW	k	Add Literal and W	1	11	111x	kkkk	kkkk	C,DC,Z	
ANDLW	k	AND Literal with W	1	11	1001	kkkk	kkkk	Z	
CALL	k	Call Subroutine	2	10	0kkk	kkkk	kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00	0000	0110	0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to Address	2	10	1kkk	kkkk	kkkk		
IORLW	k	Inclusive OR Literal with W	1	11	1000	kkkk	kkkk	Z	
MOVLW	k	Move Literal to W	1	11	00xx	kkkk	kkkk		
RETFIE	-	Return from Interrupt	2	00	0000	0000	1001		
RETLW	k	Return with Literal in W	2	11	01xx	kkkk	kkkk		
RETURN	-	Return from Subroutine	2	00	0000	0000	1000		
SLEEP	-	Go into Standby mode	1	00	0000	0110	0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from Literal	1	11	110x	kkkk	kkkk	C,DC,Z	
XORLW	k	Exclusive OR Literal with W	1	11	1010	kkkk	kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 module.

3: If Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

Description du registre INTCON

2.2.2.3 INTCON Register

The INTCON register is a readable and writable register, which contains various enable and flag bits for the TMR0 register overflow, RB port change and external RB0/INT pin interrupts.

Note:	Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

REGISTER 2-3: INTCON REGISTER (ADDRESS 0Bh, 8Bh, 10Bh, 18Bh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE**: Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **TMR0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INTE**: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INTF**: RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF**: RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state; a mismatch condition will continue to set the bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared (must be cleared in software).
0 = None of the RB7:RB4 pins have changed state

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown