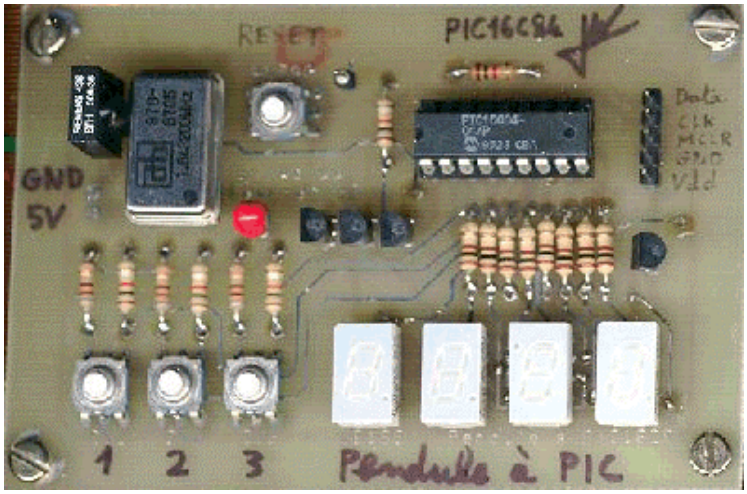
Clock' à PIC



Cet exemple est une variante de l'application AN590 de Microchip; la différence réside dans le type d'afficheurs 7 segments utilisés (anode commune).

J'ai essayé de commenter le programme en français pour en améliorer la compréhension :

PRÉSENTATION DU PROGRAMME

Ce programme réalise une horloge par affichage multiplexé de 4 afficheurs 7 segments; il est prévu pour être exécuté sur un PIC16C84 cadencé à 1,8432 MHz. Le programme peut être également compilé pour un PIC16F84.

Affichage

L'affichage se fait sur 4 afficheurs LED 7 segments sous la forme :

Les mêmes segments des 4 afficheurs sont tous reliés entre eux et sont connectés au port B (a=RB1 ; b=RB2 ; c=RB3 ; d=RB4 ; e=RB5 ; f=RB6 ; g=RB7);

Les ":" sont réalisés par les virgules des 2 afficheurs du milieu (celui de droite est retourné) qui sont connectées au bit 0 du port B.

Les afficheurs sont de type anode commune (HPDSP7511) et leur anode est connectée au port A (digit0=RA3 ; digit1=RA2 ; digit2=RA1 ; digit3=RA0).

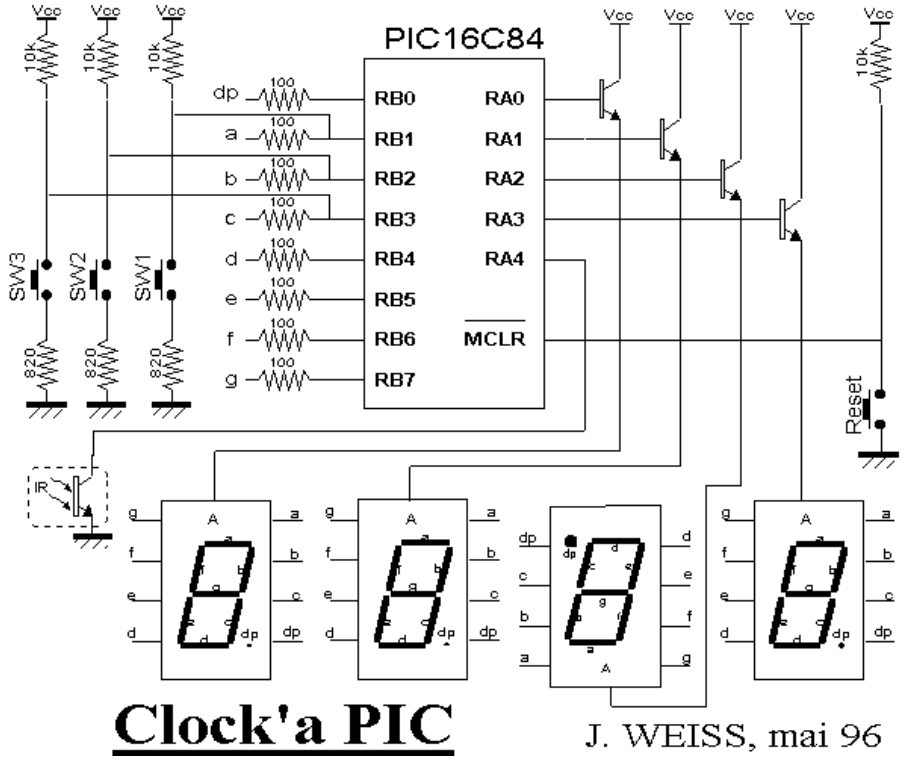
INTERRUPTEURS

Les 12 lignes d'entrée/sortie étant utilisées, il faut effectuer un multiplexage pour pouvoir insérer les interrupteurs ; cela est réalisé sur le port B (SW1=RB1 ; SW2 = RB2 ; SW3 =RB3); il faut donc que le programme commute ces lignes en entrée pour pouvoir analyser l'état des interrupteurs.

SW1 : affichage des secondes
SW2 : règlage des minutes

SW2 : règlage des heures

SCHÉMA FONCTIONNEL



L'oscillateur à quartz utilisé génère une fréquence de 1,8432 MHz, cela donne une fréquence interne de 460,8 kHz; la rotation du timer (256 états) prédivisé dans un rapport 8 se fait alors à une fréquence de 225 Hz.

Programme

```
;
;*****************************************************************
       LIST    P = 16C84
       LIST    F = INHX8M
;
;                      Clock à PIC
;*****************************************************************
;
;                      PRÉSENTATION DU PROGRAMME
;
; Ce programme tourne sur un PIC16C84.
;
;                       Description de la carte
;
;  Affichage
; L'affichage se fait sur 4 afficheurs LED 7 segments sous la forme :
;
;                      ( 88:88 )
;
; Les mêmes segments des 4 afficheurs sont tous reliés entre eux et sont
; connectés au port B (a=RB1 ; b=RB2 ; c=RB3 ; d=RB4 ; e=RB5 ; f=RB6 ; g=RB7);
;
; Les ":" sont réalisés par les virgules des 2 afficheurs du milieu (celui de
; droite est retourné) qui sont connectées au bit 0 du port B.
;
; Les afficheurs sont de type anode commune (HPDSP7511) et leurs anode est
; connectée au port A (digit0=RA3 ; digit1=RA2 ; digit2=RA1 ; digit3=RA0).
;
;  INTERRUPTEURS
; Les 12 lignes d'entrée/sortie étant utilisées, il faut effectuer un
; multiplexage pour pouvoir insérer les interrupteurs ; cela est réalisé
; sur le port B (SW1=RB1 ; SW2 = RB2 ; SW3 =RB3); il faut donc que le
; programme commute ces lignes en entrée pour pouvoir analyser l'état des
; interrupteurs.
;       SW1 : affichage des secondes
;       SW2 : règlage des minutes
;       SW2 : règlage des heures
;
; L'oscillateur à quartz utilisé génère une fréquence de 1,8432 MHz, cela
; donne une fréquence interne de 460,8 kHz; la rotation du timer (256 états)
; prédivisé dans un rapport 8 se fait alors à une fréquence de 225 Hz.
;
;       Programme        :   clkapic.ASM
;       Date de Révision :   Février 95
;
;*************************   Déclarations ************************
;
;PORT_A  equ     H'05' ; anodes communes des afficheurs 7 segments
;PORT_B  equ     H'06' ; segments et virgules des afficheurs et interrupteurs
;
ZEROT   equ     H'81'
UN      equ     H'F3'
DEUX    equ     H'49'
TROIS   equ     H'61'
QUATRE  equ     H'33'
CINQ    equ     H'25'
SIX     equ     H'05'           ; Table des segments (PORT_B)
SEPT    equ     H'F1'
HUIT    equ     H'01'
NEUF    equ     H'31'
BLANC   equ     H'FF'
; variables pour le comptage
MAXFRAC equ     D'31'           ; Timer : 256-31=225, ce qui donne 1 s
MAXSECS equ     D'196'          ; Secondes : 256-196=60
MAXMINS equ     D'196'          ; Minutes  : 256-196=60
MAXHRS  equ     D'244'          ; Heures   : 256-244=12
MINHRS  equ     D'243'
ADJMIN  equ     D'9'            ; nombre à soustraire chaque minute
ADJHR   equ     D'34'           ; nombre à soustraire chaque heure
ADJDAY  equ     D'3'            ; nombre à soustraire chaque 1/2 journée
;
AFF0    equ     B'00000001'
AFF1    equ     B'00000010'     ; Table de selection des afficheurs (PORT_A)
AFF2    equ     B'00000100'
AFF3    equ     B'00001000'
AFF_OFF equ     B'00000000'
SWITCH  equ     B'00001110'     ; Sélection des interrupteurs
;
```

```
;  Désignation des bits d'état
SEC     equ     0               ; bit d'état des secondes
MIN     equ     1               ; bit d'état des minutes
HRS     equ     2               ; bit d'état des heures
CHG     equ     3               ; changement d'état d'un interrupteur
SW1     equ     4               ; attribution des interrupteurs (0 si actif) :
SW2     equ     5               ; SW1 : Secondes-minutes, SW2 : heures, SW3 : mode
SW3     equ     6
SW_ON   equ     7               ; un interrupteur a été appuyé
;
;   VARIABLES
clefs   equ     H'18'           ; quel boutons a été appuyé ?
flags   equ     H'19'           ; bits  d'états : 0-SEC, 1-MIN, 2-HRS,
;                                       3-CHG, 4-SW1, 5-SW2, 6-SW3
;       equ     H'1A'           ; pas utilisé
affiche equ     H'1B'           ; quel afficheur raffraîchir ?
digit0  equ     H'1C'           ; Digit des minutes (à droite)
digit1  equ     H'1D'           ; Digit des dizaines de minutes
digit2  equ     H'1E'           ; Digit des heures
digit3  equ     H'1F'           ; Digit des dizaines de heures
;
;       Les variables de comptage sont initialisées pour correspondre
;       au comptage de l'heure ainsi, seconds démarre à 195D et passe
;       à 0 au bout de 61 coups.
;
sec_nth equ     H'20'           ; fractions de seconde
seconds equ     H'21'           ; secondes
minutes equ     H'22'           ; minutes
heures  equ     H'23'           ; heures
var     equ     H'24'           ; variable temporaire
count   equ     H'25'           ; Variable de comptage
count2  equ     H'26'           ; 2ème variable de comptage
;
;*******************************************************************************
;
;  Initialisation des ports et effacement de l'affichage
;
        INCLUDE "PIC84JW.H"

        ORG     PIC84
        GOTO    START
;
;
START   movlw   2               ; initialisation du registre option
        BSF     STATUS,RP0      ; Passage au banc mémoire 1
        movwf   OPT             ; Règlage du timer RTCC (1:16) sur horloge interne
;
        movlw   0
        movwf   TRISA           ; Tous les bits du port A en sortie
        movwf   TRISB           ; Tous les bits du port B en sortie
        BCF     STATUS,RP0      ; Retour au banc mémoire 0
        movlw   BLANC
        movwf   PORT_B          ; On n'affiche rien
;
;  initialize variables
        movlw   1
        movwf   RTCC            ; Initialisation de RTCC à 1
        movlw   AFF0
        movwf   affiche         ; Initialisation des variables d'affichage.
        movlw   BLANC           ; Effacement de tous les segments (pas d'affichage).
        movwf   digit0
        movwf   digit1
        movwf   digit2
        movwf   digit3
        movlw   MAXFRAC         ; Initialisation des variables du timer
        movwf   sec_nth
        movlw   MAXSECS
        movwf   seconds
        movlw   MAXMINS
        movwf   minutes
        movlw   0FFH            ; Les heures démarrent à 12 (FFH)
        movwf   heures
        movlw   0
        movwf   flags           ; Initialisation du mot d'état à 0
        clrwdt                  ; effacement du chien de garde (< 18 ms)
;
;
MAIN
;
;  Attente de la fin de comptage du timer
COMPTEUR
```

```
        movf    RTCC,0          ; RTCC --> w
        btfss   STATUS,Z_BIT    ; RTCC n'est pas perturbé dans son comptage
        goto    COMPTEUR
;
        incfsz  sec_nth,1       ; Incrémentation des fractions de secondes
        goto    TEMPS_OK        ; Incrémentation de l'horloge si 0
        clrwdt
        movlw   MAXFRAC
        movwf   sec_nth         ; Rechargement des fractions de secondes
;
VERIF_SW
        btfss   flags,SW_ON     ; A-t'on pressé sur un interrupteur ?
        goto    MISE_A_LHEURE   ; non alors on affiche l'heure
        btfsc   flags,SW1       ; Est-ce SW1 (affichage des secondes)
        goto    MISE_A_LHEURE    ; Oui, alors on ne change pas l'heure
        movlw   MAXSECS         ; non, alors on passe en mode règlage de l'horloge
        movwf   seconds         ; on remet les secondes à 0 à la mise à l'heure
        movlw   H'7F'           ; Accélération du timer (1/2 seconde)
        movwf   sec_nth
        btfss   flags,SW2       ; Est-ce SW2 (règlage des minutes)
        goto    REGLAGE         ; non, alors les minutes ne changent pas, on vérifie l'heure
        movlw   H'AF'           ; oui, on accélère le timer pour le règlage des minutes (1/2 s)
        movwf   sec_nth
        incfsz  minutes,1       ; incrémentation des minutes
        goto    REGLAGE         ; si pas de débordement, on règle l'heure
        movlw   MAXMINS         ; réinitialisation des minutes
        movwf   minutes
;
REGLAGE
        btfsc   flags,SW2       ; Est-ce SW2 (règlage des heures)
        goto    VERIF_TEMPS     ; non, on continue
        incfsz  heures,1        ; incrémentation des heures
        goto    VERIF_TEMPS     ; si pas de débordement, on vérifie l'horloge
        movlw   MAXHRS          ; réinitialisation des heures
        movwf   heures
        goto    VERIF_TEMPS     ; affichage des données
;
MISE_A_LHEURE
        bsf     flags,SEC       ; Les secondes, si elles sont affichées doivent être actualisées
        bsf     flags,CHG       ; il y a eu des changements
        incfsz  seconds,1       ; incrémentation des secondes
        goto    TEMPS_OK        ; il n'y a pas eu de débordement, on saute
        movlw   MAXSECS         ; Débordement => réinitialisation des secondes
        movwf   seconds
;
        bsf     flags,MIN       ; Les minutes doivent être réactualisées (si besoin)
        bsf     flags,CHG       ; il y a eu des changements
;       movlw   ADJMIN
;       subwf   sec_nth,1       ; Ajustement à chaque minute
        incfsz  minutes,1       ; Incrémentation des minutes
        goto    TEMPS_OK         ; C'est OK jusqu'à FFH (fin de comptage)
        movlw   MAXMINS         ; Réinitialisation des minutes
        movwf   minutes
;
        bsf     flags,HRS       ; les heures ont changé (bit d'état mis à 1)
        bsf     flags,CHG       ; il y a eu des changements
;       movlw   ADJHR
;       addwf   sec_nth,1       ; Ajustement à chaque heure
        incfsz  heures,1        ; Incrémentation des heures
        goto    TEMPS_OK        ; C'est OK jusqu'à FFH (fin de comptage)
        movlw   MAXHRS          ; Réinitialisation des heures
        movwf   heures
;       movlw   ADJDAY
;       subwf   sec_nth,1       ; Ajustement à chaque 1/2 journée (12h00)
;
TEMPS_OK
        btfss   flags,CHG       ; Y a-t'il eu changement ? (digits ou interrupteurs)
        goto    CYCLE           ;  non, alors on saute
;
VERIF_SECONDES
;  A-t'on appuyé sur les secondes ? (SW1)
        btfss   flags,SW1
        goto    VERIF_TEMPS
        movlw   0
        movwf   digit1  ; le 3ème digit sert à stocker temporairement la valeur hexa des heures
        movwf   digit2
        movwf   digit3
        movlw   MAXSECS
        subwf   seconds,0
        movwf   digit0  ; le premier digit sert à stocker temporairement la valeur hexa des
secondes
```

```
        goto    SPLIT_HEX
;
VERIF_TEMPS
        movlw   0
        movwf   digit3    ; On place un 0 dans les dizaines s'il n'y a pas d'incrémentation
        movwf   digit1
        movlw   MINHRS
        subwf   heures,0        ; le 3ème digit sert à stocker temporairement
        movwf   digit2          ; la valeur hexa des heures
        movlw   MAXMINS
        subwf   minutes,0       ; le premier digit sert à stocker temporairement
        movwf   digit0          ; la valeur hexa des minutes
;
SPLIT_HEX               ;  séparation des variables d'affichage en 2 mots hexa
        movlw   2
        movwf   count   ; boucle pour convertir chaque nombre en secondes, minutes et heures
;
;                       1er passage : FSR = digit0, 2ème passage : FSR = digit2
;
        movlw   digit0          ; adresse du digit0 dans File Select Register
        movwf   FSR             ; et validation de POINTER
        goto    BOUCLE          ; Déplacement des minutes/secondes
;
BOUCLE2 movlw   digit2
        movwf   FSR             ; Déplacement des heures
;
BOUCLE
        movlw   D'10'
        subwf   POINTER,1       ; Comptage du nombre de dizaines
        btfsc   STATUS,CARRY    ; Il y a-t'il débordement (borrow)
        goto    INCREMENT_10S   ; si non, on ajoute 1 à la position des dizaines
        addwf   POINTER,1       ; si oui, on ajoute 10 pour obtenir 1 seconde
        goto    NEXT_DIGIT
;
INCREMENT_10S
        incf    FSR,1     ; bump address pointed to from 1s positoion to 10s
        incf    POINTER,1 ; add 1 to 10s position as determined by previous subtract
        decf    FSR,1     ; put POINTER value back to 1s place for next subtraction
        goto     BOUCLE   ; go back and keep subtracting until finished
;
NEXT_DIGIT
        decfsz  count,1
        goto    BOUCLE2
;
CONVERT_HEX_TO_DISPLAY          ; conversion hexa -> décimal
        movlw   digit0
        movwf   FSR             ; Adresse du 1er digit dans FSR pour valider POINTER
        movlw   H'4'
        movwf   count           ; Il y 4 afficheurs à gérer
NEXT_HEX
        movf    POINTER,0       ; Récupération de la valeur du digit (Hexa)
        call    RENVOI_CODE     ; Recherche du code dans la table des segments
        movwf   POINTER         ; Ecriture dans la variable digit
        incf    FSR,1           ; Pointage vers le digit suivant
        decfsz  count,1         ; A-t'on fait les 4 digits
        goto    NEXT_HEX
;
FIX_DISPLAY
        movlw   ZEROT
        subwf   digit3,0
        btfss   STATUS,Z_BIT    ; les 4ème digit est-il nul ?
        goto    FIX_SEC         ; non, alors on l'affiche
        movlw   BLANC           ; oui, alors on le masque
        movwf   digit3

FIX_SEC
        btfss   flags,SW1       ; Est-on en mode d'affichage des secondes ?
        goto    CLEAR_FLAGS     ; non, on  saute
        movwf   digit2          ; oui, alors on n'affiche pas le 3ème digit
;
CLEAR_FLAGS
        movlw   H'F0'
        andwf   flags,1         ; Effacement des 4 bits d'état de poids faible
;
CYCLE
        movlw   AFF_OFF
        movwf   PORT_A          ; Extinction des afficheurs
        movlw   SWITCH
        BSF     STATUS,RP0      ; Passage au banc mémoire 1
        movwf    TRISB          ; Bits du port B en entrée (interrupteurs)
        BCF     STATUS,RP0       ; Retour au banc mémoire 0
```

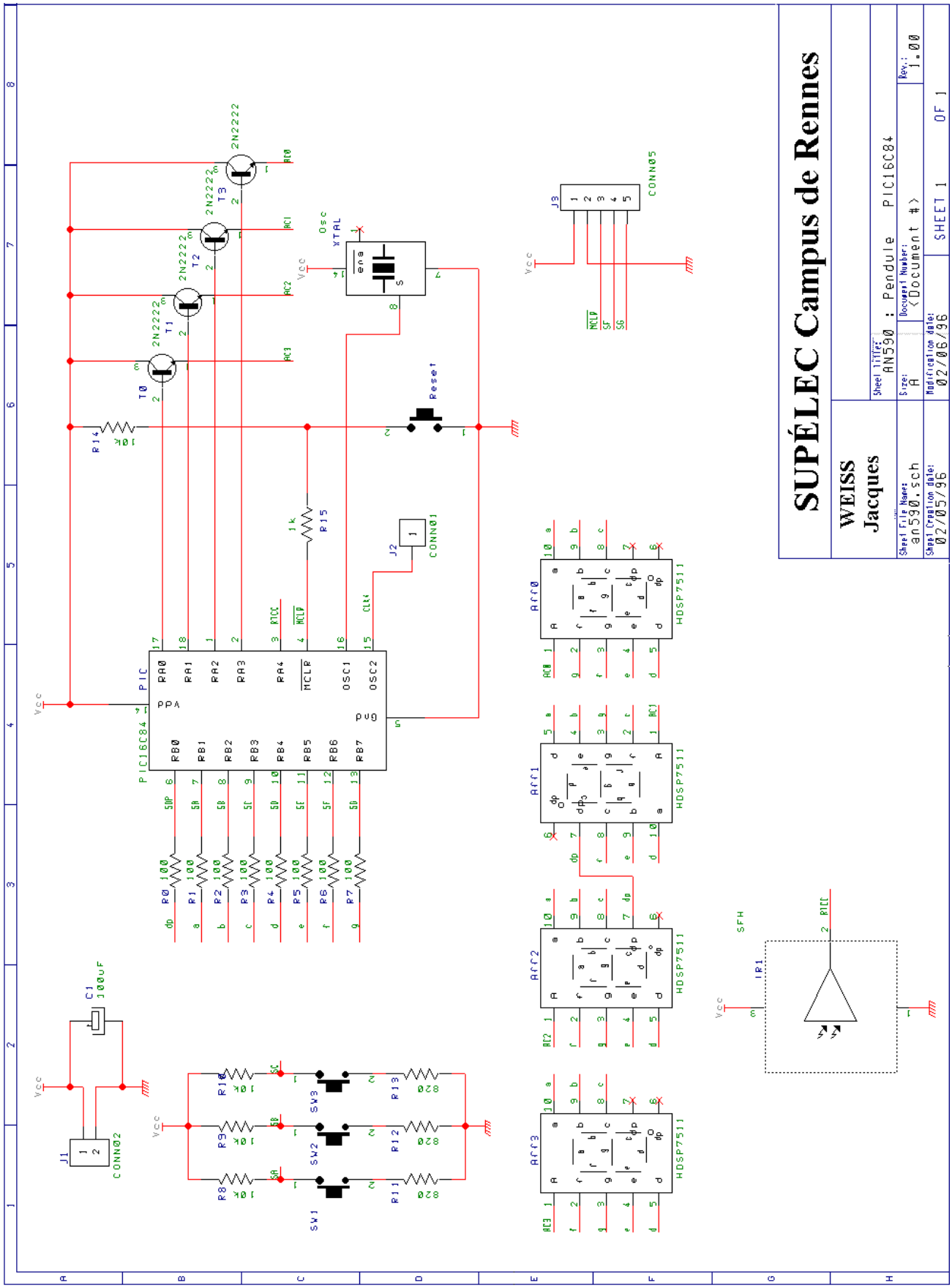
```
        movlw   H'0F'
        nop
        nop
        nop
        nop
        nop
        andwf   flags,1         ; Réinitialisation du mot d'état
        movf    PORT_B,0
        xorlw   H'FF'           ; Complémentation des données
        movwf   var
        btfss   var,1
        goto    SWITCH2
        bsf     flags,CHG
        bsf     flags,SW1
        bsf     flags,SW_ON
SWITCH2 btfss   var,2
        goto    SWITCH3
        bsf     flags,CHG
        bsf     flags,SW2
        bsf     flags,SW_ON
SWITCH3 btfss   var,3
        goto    SETPORT
        bsf     flags,CHG
        bsf     flags,SW3
        bsf     flags,SW_ON
;
SETPORT
        movlw   0
        BSF     STATUS,RP0      ; Passage au banc mémoire 1
        movwf   TRISB
        BCF     STATUS,RP0      ; Retour au banc mémoire 0
        movlw   BLANC
        movwf   PORT_B
;
;   détermination de l'afficheur à réactualiser
;
        btfsc   affiche,0       ; Est-ce le premier digit ?
        movf    digit3,0        ; si oui, on le place dans w
        btfsc   affiche,1       ; Est-ce le deuxième digit ?
        movf    digit2,0        ; si oui, on le place dans w
        btfsc   affiche,2       ; Est-ce le troisième digit ?
        movf    digit1,0        ; si oui, on le place dans w
        btfsc   affiche,3       ; Est-ce le quatrième digit ?
        movf    digit0,0        ; si oui, on le place dans w
        movwf   PORT_B          ; on envoie w vers les afficheurs (Port B)
        btfsc   sec_nth,7
        bcf     PORT_B,0        ; Clignotement des ":" (50 % du temps)
        movf    affiche,0       ; Cycle d'affichage du digit concerné
        movwf   PORT_A          ; Validation de l'affichage
        rlf     affiche,1       ; Rotation de l'affichage
        bcf     affiche,0       ; on place un 0 dans le LSB
        btfsc   affiche,4       ; A-t'on réactualisé les 4 digits
        bsf     affiche,0       ; Si oui on réinitialise la variable affiche
        clrwdt
;
        goto    MAIN
;
RENVOI_CODE
        addwf   PC,1
        retlw   ZEROT
        retlw   UN
        retlw   DEUX
        retlw   TROIS
        retlw   QUATRE
        retlw   CINQ
        retlw   SIX
        retlw   SEPT
        retlw   HUIT
        retlw   NEUF

    END
```

Fichier PIC84JW.H

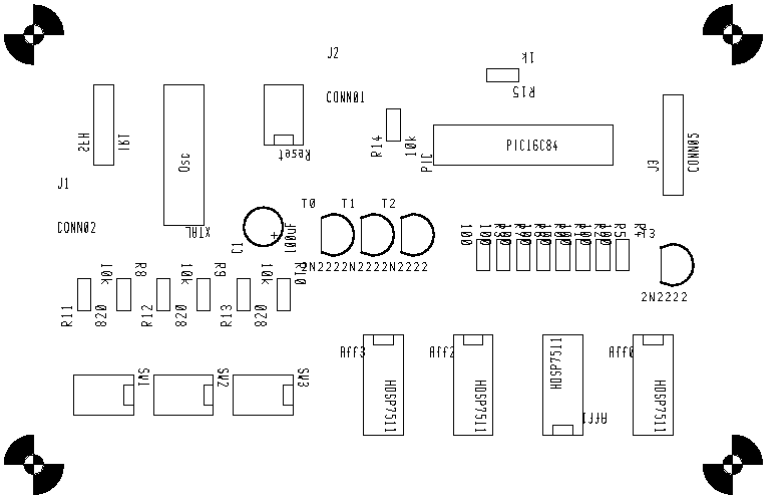
```
       LIST    P = 16C84
       LIST   F = INHX8M
       __config       H'3F1A' ; CP : off,PWRTE : on, WDT : off, HS OSC

;NAME.........................VALUE............TYPE
;-----------------------------------------------------

PIC84                  EQU     000H           ;RESET VECTOR
RAM                    equ     0CH            ; départ de la RAM
EEPROM                 equ     2100H          ; départ de l'EEPROM
RETOUR_INT             equ     004H
CONFIG                 equ     2007H
POINTER                equ     00H
FSR                    equ      4H
OPT                    equ      1H
RTCC                   EQU      1H            ;
PC                     EQU      2H            ;

STATUS                 EQU      3H              ;F3 REG IS STATUS REG.
; STATUS REG. BITS
CARRY                  EQU      0H            ;       CARRY
D_CARRY                EQU      1H            ;       DIGITAL CARRY
Z_BIT                  EQU      2H            ;       ZERO
P_DOWN                 EQU      3H            ;       POWER DOWN
T_OUT                  EQU      4H            ;       T_OUT
RP0                    EQU      5H            ;STATUS BITS
RP1                    EQU      6H            ;STATUS BITS
IRP                    EQU      7H            ;STATUS BITS

INTCON                 EQU     00BH           ;INTCON REGISTER
GIE                    EQU      7H            ;INTCON REGISTER BITS
EEIE                   EQU      6H            ;INTCON REGISTER BITS
T0IE                   EQU      5H            ;INTCON REGISTER BITS
INTE                   EQU      4H            ;INTCON REGISTER BITS
RBIE                   EQU      3H            ;INTCON REGISTER BITS
RTIF                   EQU      2H            ;INTCON REGISTER BITS
INTF                   EQU      1H            ;INTCON REGISTER BITS
RBIF                   EQU      0H            ;INTCON REGISTER BITS
;
FSR                    EQU      4H            ;
;
PORT_A                 EQU      5H
PORT_B                 EQU      6H              ; I/O PORT ASSIGNMENTS
;
TRISA                  EQU      05H           ;SPECIAL-PURPOSE REGISTERS
TRISB                  EQU      06H           ;SPECIAL-PURPOSE REGISTERS
;
W                      EQU     0H
F                      EQU     1H

EEDATA                 EQU     8H             ;EEPROM DATA REGISTER
EEADR                  EQU     9H             ;EEPROM ADDRESS REGISTER

EECON1                 EQU     08H            ;EEPROM Control Register 1
EEIF                   EQU     4H             ;EECON1 REGISTER BITS
WRERR                  EQU     3H             ;EECON1 REGISTER BITS
WREN                   EQU     2H             ;EECON1 REGISTER BITS
WR                     EQU     1H             ;EECON1 REGISTER BITS
RD                     EQU     0H             ;EECON1 REGISTER BITS

EECON2                 EQU     09H            ;EEPROM Control Register 2
```

Maquette

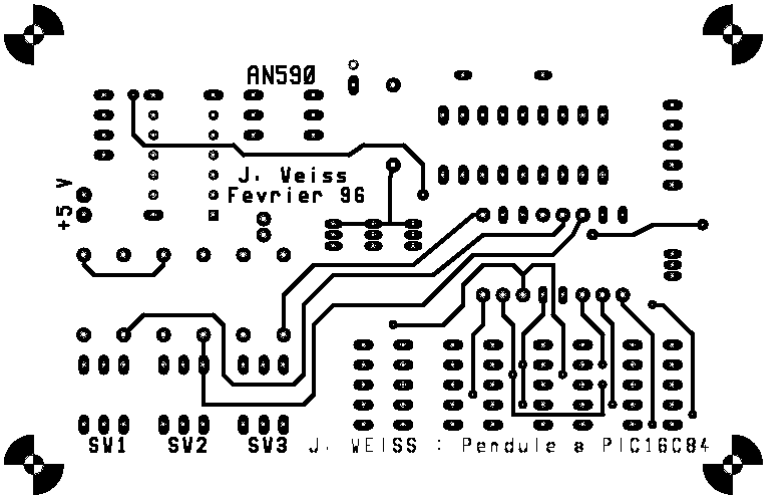
Schéma électrique



Implantation



côté composants



côté cuivre