CONTENIDO

A) LISTADO DEL PROGRAMA

B) MANUAL DEL USUARIO

LISTADO DEL PROGRAMA

E.M.C.A.

```
;*******************************************************************************
;***     Programa para control del Equipo de Monitoreo  de Contaminacion

;***     del Aire por Escapes

;***     E.M.C.A.

;***     noviembre de 1999

;***     Programa para microcontrolador DS2250T

;***     Compilado con CYS8051

;*******************************************************************************

;       El sistema controla:

;       1.-   8 Entradas analogicas convertidas con ADC0808
;             con rango completo de 0 a +5VDC

;       2.-   1 display de una linea de 16 caracteres

;       3.-   1 Teclado de 16 teclas

;       4.-   1 Interface RS-232C
;
;*******************************************************************************
;><><><><><><><><><><.............................><><><><><><><><><><><>
;*******************************************************************************
;       A C T U A L I Z A C I O N E S
;
;
;*******************************************************************************
;><><><><><><><><><><.............................><><><><><><><><><><><>
;*******************************************************************************
;
;
;       AREA DE DEFINICION DE   EQUIVALENTES
;
;

datos_conv  equ   P1    ;Bus de datos para el conversor

port_dis    equ   P0    ;Bus de datos general de display

disp_b7         equ     P0.3    ;Bus de datos display bit 7
disp_b6         equ     P0.2    ;Bus de datos display bit 6
disp_b5         equ     P0.1    ;Bus de datos display bit 5
disp_b4         equ     P0.0    ;Bus de datos display bit 4

ena             equ     P0.4    ;Control ENABLE display
r_w             equ     P0.5    ;Control READ/WRITE display
rs              equ     P0.6    ;Control RESET display


out_ena         equ     T0      ;Habilita salidas del conversor
start           equ     P2.7    ;Control de arranque del conversor
```

```
add_2       equ   P2.6  ;Direccion A2 del conversor
add_1       equ   P2.5  ;Direccion A1 del conversor
add_0       equ   P2.4  ;Direccion A0 del conversor

end_of_con  equ   INT0              ;End of conversion del ADC0808

teclas      equ     P2          ;Datos de teclado en bits P2.0 a P2.3
tecla_int   equ   INT1


;
;*******************************************************************************
;
;     EQUIVALENTES DEL TECLADO
;
;*******************************************************************************
;

uno         equ   00H
dos         equ   01H
tres        equ   03H
sube        equ   02H

cuatro      equ   04H
cinco       equ   05H
seis        equ   07H
baja        equ   06H

siete       equ   08H
ocho        equ   09H
nueve       equ   0BH
s2nd        equ   0AH

clear       equ   0CH
cero        equ   0DH
help        equ   0FH
enter       equ   0EH

;
;*******************************************************************************
;
;     AREA DE DEFINICION DE CONSTANTES
;
;*******************************************************************************
;


MCON            equ     0C6H
TA              equ     0C7H
stack           equ     2FH
num_car         equ     8           ;Numero de caracteres que controla el
                                    ;monitor del display
car_ret         equ     13
```

```
;
;*******************************************************************************
;
;     AREA DE LOCALIZACION DE RAM
;
;*******************************************************************************
;
dato_analogo        equ     40H
data_dis            equ     41H
cien                equ     42H
dato_tecla          equ     43H     ;Registro de almacenamiento de ultimo
                                    ;dato del teclado
convierte           equ     44H
seg_dat             equ     45H     ;direccion de registro de segundos
min_dat             equ     46H     ;direccion de registro de minutos
hora_dat            equ     47H     ;direccion de registro de horas
dias_dat            equ     48H     ;direccion de registro de dia de la semana
diam_dat            equ     49H     ;direccion de registro de dia del mes
mese_dat            equ     4AH     ;direccion de registro del mes
anio_dat            equ     4BH     ;direccion de registro del ano

espera              equ     4CH
dato_leido          equ     4DH
hay_tecla           equ     4EH
funcion             equ     4FH
inicio_d            equ     50H     ;Area de RAM para manejo de display
                                    ;hasta  5FH
dato_mono1          equ     60H
dato_mono2          equ     61H
dato_hume1          equ     62H
dato_hume2          equ     63H
dato_temp1          equ     64H
dato_temp2          equ     65H


;
;*******************************************************************************
;
;*******************************************************************************
;
;     Direccion de inicio del programa con HARDWARE - RESET

      org   0H

h_reset:

        ljmp    inicio


;
;
;*******************************************************************************
;
;     Area de entrada de interrupciones generales
;
;*******************************************************************************
```

```
        org   03H
        ljmp  fin_conversion    ;Va a rutina de lectura del conversor

        org     13H             ;Origen de interrupcion del TECLADO
        ljmp  teclado           ;Va a rutina de atencion al teclado

        ORG     23H
        ljmp    com_serial      ;interrupcion de comunicacion serial
;
;
;*******************************************************************************
;
;
        org   100H

inicio:

        mov     SP,#stack       ;Inicializa el SP con STACK(30)
        mov     TA,#0AAH        ;Inicializacion para acceso temporizado
        mov     TA,#55H
        mov     PCON,#0         ;pone en reset todos los parametros de
                                ;control
                                ;Al final esta inicio de RAM externa en
                                ;1800H. Cuando inicializa el reloj
                                ;y registros especiales tambien inicializa
                                ;el DPTR
        mov   MCON,#38H         ;selecciona direccion de inicio de RAM en
                                ;1800H y 32 K total en el chip
                                ;No mas de 4096 en programa
        mov     IE,#0           ;Inicializa control de interrupciones
        mov   TMOD,#21H         ;Inicializa la puerta serial
        mov   TH1,#0F8H         ;     2400 BAUD
        ORL   PCON,#88H         ;Pone en doble baud_rate
                               ;NO PARITY
                               ;1 stop bit
        mov   SCON,#50H         ;Pone la puerta serial en modo 0
        mov   TCON,#40H

          SETB    ES           ;habilita la interrupcion serial
        mov   P2,#0FH

          lcall   inicio_display

          setb    EA              ;habilita las interrupciones
                                  ;generales

          mov     dato_tecla,#0   ;Encera registro de teclado para
                                  ;conversor
          mov     hay_tecla,#'N'

          lcall titulos_inicio
          lcall   del_1seg

;
;*******************************************************************************
;
```

```
prog_prin:

      lcall titulos_rut
      lcall del_1seg

      mov   A,hay_tecla
      cjne  A,#'S',sigue_1
      ljmp  ver_teclas

;
;*******************************************************************************
;

sigue_1:

      setb  EX1
      lcall       reloj
      lcall display
      mov   A,hay_tecla
      cjne  A,#'S',sigue_2
      ljmp  ver_teclas
;
;*******************************************************************************
;

sigue_2:

      lcall del_1seg
      mov   A,hay_tecla
      cjne  A,#'S',sigue_3
      ljmp  ver_teclas
;
;*******************************************************************************
;

sigue_3:

      mov         DPTR,#registra
      movx        A,@DPTR
      cjne        A,#'F',ver_temperatura
      ljmp        medir_temperatura

ver_temperatura:

      cjne  A,#'T',no_reg_temp

medir_temperatura:

      mov   dato_tecla,#cero
      mov   funcion,#'T'

      lcall   lee_solo

      lcall valor_tempera

      lcall solo_tem_dis

      mov     A,dato_temp1          ;Trae valor de lectura de la
```

```
        anl     A,#0FH                  ;temperatura y lo pone en modo Ascii
        orl     A,#30H
        mov     R0,#(inicio_d+8)        ;Ubica en el display el primer valor
        mov     @R0,A

        mov     A,dato_temp2
        swap    A
        anl     A,#0FH
        orl     A,#30H
        inc     R0
        mov     @R0,A
        inc     R0
        mov     @R0,#','
        mov     A,dato_temp2
        anl     A,#0FH
        orl     A,#30H
        inc     R0
        mov     @R0,A

        lcall display
        lcall del_lseg

;
;*****************************************************************************
;

        mov     DPTR,#registra
        movx    A,@DPTR

no_reg_temp:

          cjne    A,#'F',ver_humedad
          ljmp    medir_humedad

ver_humedad:

        cjne  A,#'H',no_reg_hum

medir_humedad:

        mov     dato_tecla,#uno
        mov     funcion,#'H'

        lcall   lee_solo

        lcall valor_humedad

        lcall solo_hum_dis

        mov     A,dato_humel
        swap    A
        anl     A,#0FH
        jz      no_cerol
        orl     A,#30H
        mov     R0,#(inicio_d+8)
        mov     @R0,A

no_cerol:
```

```
        mov     A,dato_hume1            ;Trae valor de lectura de humedad
        anl     A,#0FH                  ;Lo pone en modo Ascii
        orl     A,#30H
        mov     R0,#(inicio_d+9)        ;Ubica en el display el segundo valor
        mov     @R0,A
        mov     A,dato_hume2
        swap    A
        anl     A,#0FH
        orl     A,#30H
        inc     R0
        mov     @R0,A
        inc      R0
        mov     @R0,#','
        mov     A,dato_hume2
        anl     A,#0FH
        orl     A,#30H
        inc     R0
        mov     @R0,A

      lcall display
      lcall del_1seg

;
;*****************************************************************************
;

      mov   DPTR,#registra
      movx  A,@DPTR

no_reg_hum:

        cjne    A,#'F',ver_monoxido
        ljmp    medir_monoxido

ver_monoxido:
      cjne  A,#'M',no_reg_mon

medir_monoxido:

      mov   dato_tecla,#dos
      mov   funcion,#'M'

      lcall   lee_solo

      lcall valor_mono

      lcall solo_mon_dis

      mov     A,dato_mono1
      swap    A
      anl     A,#0FH
      orl     A,#30H
      mov     R0,#(inicio_d+8)        ;Ubica en el display el primer valor
      mov     @R0,A
      mov     A,dato_mono1            ;Trae valor de lectura de monoxido
      anl     A,#0FH                  ;Lo pone en modo Ascii
      orl     A,#30H
      inc     R0
      mov     @R0,A
```

```
      mov     A,dato_mono2
      swap    A
      anl     A,#0FH
      orl     A,#30H
      inc     R0
      mov     @R0,A
      mov     A,dato_mono2
      anl     A,#0FH
      orl   A,#30H
      inc   R0
      mov   @R0,A

      lcall display
      lcall  del_1seg

      ljmp    alterno_3

;
;*******************************************************************************
;

no_reg_mon:

        cjne      A,#'F',no_registros
        ljmp    alterno_3

;
;*******************************************************************************
;

no_registros:

      lcall no_registro_dis
        ljmp    alterno_3

;
;*******************************************************************************
;

alterno_3:

      mov   A,hay_tecla
      cjne  A,#'S',sigue_4
      ljmp  ver_teclas
;
;
;*******************************************************************************
;

sigue_4:

      lcall       intervalo_dis

      mov   DPTR,#inter_dec
      movx  A,@DPTR
      mov   B,A
      mov   R0,#(inicio_d+9)
      swap  A
      anl   A,#0FH
```

```
        orl     A,#30H
        mov     @R0,A
        mov     A,B
        mov     R0,#(inicio_d+10)
        anl     A,#0FH
        orl     A,#30H
        mov     @R0,A

        lcall         display

        mov     A,hay_tecla
        cjne    A,#'S',sigue_5
        ljmp    ver_teclas
;
;*******************************************************************************
;
sigue_5:

        lcall del_1seg

        mov     A,hay_tecla
        cjne    A,#'S',sigue_6

        ljmp    ver_teclas

;
;*******************************************************************************
;
sigue_6:

        mov       A,min_dat             ;Verifica si el ultimo registro se realizo
                                        ;en el mismo minuto actual
        mov     B,A
        mov     DPTR,#ultimo_reg
        movx    A,@DPTR
        xrl     A,B
        jz      no_ver_registro
        mov       A,min_dat               ;Actualiza el ultimo registro
        movx    @DPTR,A
        mov       DPTR,#minutos_r         ;Actualiza el contador de minutos
        movx    A,@DPTR
        inc     A
        movx    @DPTR,A
        mov     B,A
        mov       DPTR,#intervalo         ;Verifica si ya llego al valor del
        movx      A,@DPTR                 ;intervalo para registrar las entradas
        xrl     A,B
        jnz     no_ver_registro
        lcall     leer_entradas           ;Va a rutina de leer y grabar los
        mov       DPTR,#minutos_r         ;datos leidos
        mov       A,#0H                   ;Encera el contador de minutos
       movx   @DPTR,A

no_ver_registro:

         ljmp     prog_prin


;
```

```
;
;*****************************************************************************
;                  AREA  DE  SUBRUTINAS
;*****************************************************************************
;


titulos_inicio:

      lcall   epn_disp              ;Muestra en display  REGISTROS
      lcall       display
      lcall del_1seg

      lcall   titulo2
      lcall       display
      lcall del_1seg

      lcall   titulo3
      lcall       display
      lcall del_1seg

      lcall   titulo4
      lcall       display
      lcall del_1seg

      lcall   titulo5
      lcall       display
      lcall del_1seg

        ret

titulos_rut:

      lcall   titulo6
      lcall       display
      lcall del_1seg

      lcall   titulo7
      lcall       display
      lcall del_1seg


      lcall   regis_disp
      lcall       display

        ret

;
;*****************************************************************************
;
;     Rutina para leer las entradas del conversor. Dependiendo que
;     registros se hayan programado
;

leer_entradas:

      mov   DPTR,#registra
      movx  A,@DPTR
      cjne  A,#'T',no_mid_temp
```

```
        mov   dato_tecla,#cero
        mov   funcion,#'T'

        lcall   lee_guardar
        ret

no_mid_temp:

        cjne  A,#'H',no_mid_hume
        mov   dato_tecla,#uno
        mov   funcion,#'H'

        lcall   lee_guardar
        ret

no_mid_hume:

        cjne  A,#'M',no_mid_mono
        mov   dato_tecla,#dos
        mov   funcion,#'M'
        lcall   lee_guardar
        ret

no_mid_mono:

        cjne  A,#'F',no_mid_nada
        mov   dato_tecla,#cero
        mov   funcion,#'T'
        lcall   lee_guardar
        mov   dato_tecla,#uno
        mov   funcion,#'H'
        lcall   lee_guardar
        mov   dato_tecla,#dos
        mov   funcion,#'M'
        lcall   lee_guardar

        mov   funcion,#'F'
        ret

no_mid_nada:
        ret

;
;*******************************************************************************
;

lee_solo:

        lcall leer_conversor
        ret

;
;*******************************************************************************
;
```

```
lee_guardar:

      lcall leer_conversor

      mov   DPTR,#direcc_low
      movx  A,@DPTR                    ;Actualiza la direccion del DPTR
      push  ACC
      mov   DPTR,#direcc_hig
      movx  A,@DPTR
      mov   DPH,A
      pop   ACC
      mov     DPL,A                     ;para guardar el dato
                                        ;temperatura, humedad, monoxido
      mov   A,funcion
      movx  @DPTR,A
      mov   A,anio_dat
      inc   DPTR
      movx  @DPTR,A
      mov   A,mese_dat
      inc   DPTR
      movx  @DPTR,A
      mov   A,diam_dat
      inc   DPTR
      movx  @DPTR,A
      mov   A,hora_dat
      inc   DPTR
      movx  @DPTR,A
      mov   A,min_dat
      inc   DPTR
      movx  @DPTR,A
      mov   A,dato_analogo
      inc   DPTR
      movx  @DPTR,A
      inc   DPTR
      mov   A,#'Z'                      ;Este valor indica el fin del
      movx  @DPTR,A                     ;del archivo de datos
      mov   A,DPL
      push  ACC
      mov   A,DPH
      mov   DPTR,#direcc_hig            ;Actualiza los pointers
      movx  @DPTR,A
      pop   ACC
      mov   DPTR,#direcc_low
      movx  @DPTR,A
;
;*******************************************************************************
;
;     Rutina para verificar que no se exedan las grabaciones de la
;     direccion 7Fxxh que es la maxima capacidad del controlador.
;     Se dejan 255 bytes libres para efectos de seguridad
;
      mov     DPTR,#direcc_hig              ;Actualiza los pointers
      movx  A,@DPTR
      cjne  A,#7FH,no_lleno

      lcall memoria_full
      lcall  display

        sjmp      $
```

```
;
;*******************************************************************************
;

no_lleno:
        ret

;
;*******************************************************************************
;
;       Rutina para admnitir teclas de programacion. Debe ingresar primero
;       la clave <2ND>123<ENTER>
;

ver_teclas:

        mov hay_tecla,#'N'
      mov   A,dato_tecla
      cjne  A,#s2nd,no_clave
      lcall del_5seg
      mov   A,hay_tecla
      cjne  A,#'S',no_clave

      mov   hay_tecla,#'N'
      mov   A,dato_tecla
      cjne  A,#uno,no_clave
      lcall del_5seg
      mov   A,hay_tecla
      cjne  A,#'S',no_clave

      mov   hay_tecla,#'N'
      mov   A,dato_tecla
      cjne  A,#dos,no_clave
      lcall del_5seg
      mov   A,hay_tecla
      cjne  A,#'S',no_clave

      mov   hay_tecla,#'N'
      mov   A,dato_tecla
      cjne  A,#tres,no_clave
      lcall del_5seg
      mov   A,hay_tecla
      cjne  A,#'S',no_clave

      mov   hay_tecla,#'N'
      mov   A,dato_tecla
      cjne  A,#enter,opcion_2

      ljmp  clave_correcta

;
;*******************************************************************************
;
```

```
opcion_2:

      cjne  A,#help,no_clave
      ljmp  prueba_conversor

no_clave:

        ljmp    prog_prin
;
;*****************************************************************************
;
clave_correcta:

      lcall   opcion_1_dis          ;Muestra en el display "Interva. ** min"

      lcall display
      lcall del_5seg

      mov   A,hay_tecla
      cjne  A,#'S',no_clave
      lcall dato_numero
      cjne  A,#'N',si_numero

      ljmp  clave_correcta

;
;*****************************************************************************
;

si_numero:

      mov   R0,#(inicio_d+9)
      mov   DPTR,#inter_dec
      anl   A,#0FH
      swap  A
      movx  @DPTR,A
      swap  A
      orl   A,#30H
      mov   @R0,A                   ;Pone en display numero mas signi-
                                    ;ficativo del intervalo
      anl   A,#0FH                  ;Prepara numero MSB para guardarlo
      mov   B,#10                   ;el digito MSB debe multiplicar por
      mul   AB                      ;16 para transformalo en hex
      mov   R5,A
      clr   C
      clr   AC

        lcall     display
        lcall  del_5seg             ;Espera por el siguiente digito

      mov   A,hay_tecla
      cjne  A,#'S',no_clave
      lcall dato_numero
      cjne  A,#'N',si_numero1

        ljmp      clave_correcta
;
;*****************************************************************************
;
```

```
si_numero1:

      mov   R0,#(inicio_d+10)
      push  ACC
      anl   A,#0FH
      mov   B,A
      mov   DPTR,#inter_dec          ;Direccion para guardar el dato del
      movx  A,@DPTR                  ;intervalo en decimal
      add   A,B
      movx  @DPTR,A
      pop   ACC
      mov   @R0,A                    ;Pone en display numero mas signi-
                                     ;ficativo del intervalo
      anl   A,#0FH                   ;Prepara numero LSB para guardarlo
      add   A,R5
      mov   DPTR,#intervalo          ;Guarda el valor hexadecimal del
      movx  @DPTR,A                  ;intervalo

        lcall     display

;
;*****************************************************************************
;
;     Luego de guardar dato de intervalo debe pedir dato de magnitudes
;     a registrar: 1 ==> Temperatura
;                  2 ==> Humedad
;                  3 ==> Monoxido
;                  4 ==> Todas
;

magnitudes:

      lcall magnitudes_dis

      lcall display

      mov   hay_tecla,#'N'

      lcall del_5seg

      mov   A,hay_tecla
      cjne  A,#'S',no_magnitud

      mov   A,dato_tecla
      cjne    A,#uno,no_temp         ;Si se digita la tecla 1 registra solo

      mov     DPTR,#registra         ;temperatura
      mov   A,#'T'
      movx  @DPTR,A

      ljmp  fin_magnitud

;
;*****************************************************************************
;
```

```
no_temp:

        cjne    A,#dos,no_hume          ;Si se digita la tecla 2 registra

        mov     DPTR,#registra          ;solo humedad
        mov     A,#'H'
        movx    @DPTR,A

      ljmp  fin_magnitud
;
;*****************************************************************************
;
no_hume:

      cjne    A,#tres,no_mono         ;Si se digita la tecla 3 registra
      mov     DPTR,#registra          ;solo monoxido  de carbono
      mov     A,#'M'
      movx    @DPTR,A
      ljmp  fin_magnitud
;
;*****************************************************************************
;
no_mono:

        cjne      A,#cuatro,no_magnitud

        mov     DPTR,#registra          ; Si se digita la tecla 4 registra
        mov     A,#'F'                  ; la temperatura, humedad y monoxido
        movx      @DPTR,A
        ljmp      fin_magnitud
;
;*****************************************************************************
;
no_magnitud:

        ljmp    prog_prin
;
;*****************************************************************************
;
fin_magnitud:

        ljmp    prog_prin
;
;*****************************************************************************
;
;
dato_numero:

      mov   A,dato_tecla        ;Rutina para transformar dato de teclado
                                ;en dato numerico y ademas eliminar datos
                                ;que no sean numericos
      mov   B,A
        cjne    A,#uno,ver_alternol
      mov   A,#31H
      ljmp  fin_dato

ver_alternol:

        cjne    A,#dos,ver_alterno2
```

```
        mov   A,#32H
        ljmp  fin_dato

ver_alterno2:

        cjne    A,#tres,ver_alterno3
        mov     A,#33H
        ljmp  fin_dato

ver_alterno3:

        cjne    A,#cuatro,ver_alterno4
        mov     A,#34H
        ljmp  fin_dato

ver_alterno4:

        cjne    A,#cinco,ver_alterno5
        mov     A,#35H
        ljmp  fin_dato

ver_alterno5:

        cjne    A,#seis,ver_alterno6
        mov     A,#36H
        ljmp  fin_dato

ver_alterno6:

        cjne    A,#siete,ver_alterno7
        mov     A,#37H
        ljmp  fin_dato

ver_alterno7:

        cjne    A,#ocho,ver_alterno8
        mov     A,#38H
        ljmp  fin_dato

ver_alterno8:

        cjne    A,#nueve,ver_alterno9
        mov     A,#39H
        ljmp  fin_dato

ver_alterno9:

        cjne  A,#cero,no_numerico
        mov   A,#30H
        ljmp  fin_dato

no_numerico:

        mov   A,#'N'

fin_dato:

        ret
```

```
;
;*******************************************************************************
;
;

prueba_conversor:

      lcall conversor_nro_dis

      lcall display
      lcall del_5seg

      mov   A,hay_tecla
      cjne  A,#'S',no_prueba
      clr   EX1
      mov   R2,#10

lazo_prueba_con:

      lcall leer_conversor          ;Va a rutina de lectura del conversor
      djnz  R2,lazo_prueba_con


no_prueba:

        ljmp    prog_prin

;
;*******************************************************************************
;
;     Rutina para leer datos de entradas analogicas
;

leer_conversor:

      mov   A,dato_tecla
        cjne    A,#cero,entrada_uno
        clr     add_0                 ;Pone direccion de control del
        clr     add_1                 ;conversor para escoger la entrada
        clr     add_2                 ;adecuada
      mov   convierte,#30H
      ljmp  sigue_leer

entrada_uno:

        cjne    A,#uno,entrada_dos
        setb    add_0
        clr     add_1
        clr     add_2
      mov   convierte,#31H
      ljmp  sigue_leer

entrada_dos:

        cjne    A,#dos,entrada_tres
        clr     add_0
        setb    add_1
        clr     add_2
```

```
;
        mov     convierte,#32H
        ljmp    sigue_leer

entrada_tres:

        cjne    A,#tres,entrada_cuatro
        setb    add_0
        setb    add_1
        clr     add_2
        mov     convierte,#33H
        ljmp    sigue_leer

entrada_cuatro:

        cjne    A,#cuatro,entrada_cinco
        clr     add_0
        clr     add_1
        setb    add_2
        mov     convierte,#34H
        ljmp    sigue_leer

entrada_cinco:

        cjne    A,#cinco,entrada_seis
        setb    add_0
        clr     add_1
        setb    add_2
        mov     convierte,#35H
        ljmp    sigue_leer

entrada_seis:

        cjne    A,#seis,entrada_siete
        clr     add_0
        setb    add_1
        setb    add_2
        mov     convierte,#36H
        ljmp    sigue_leer

entrada_siete:

        cjne    A,#siete,falla
        setb    add_0
        setb    add_1
        setb    add_2
        mov     convierte,#37H
        ljmp    sigue_leer

falla:
        lcall falla_dis

        lcall display
        lcall del_1seg              ;Llama a rutina de retardo de 1 segundo

        ret

;
;*****************************************************************************
;
```

```
sigue_leer:

        lcall   conversor_dis           ;Muestra en dipslay letrero

        mov R0,#(inicio_d+7)            ;Pone en el display el numero
                                        ;del canal de conversion escojido
        mov A,convierte
        mov @R0,A

        lcall    display

;
;*****************************************************************************
;
;     Para estabilizar la lectura del conversor realiza dos lecturas
;     descarta la primera y deja la segunda como buena
;

        setb    start           ;Arranca al conversor generando un pulso
        nop                     ;en el START/ALE
        clr     start           ;Termina el pulso

        setb    EX0             ;Habilita la interrupcion de fin de
        clr     C               ;conversion del conversor
        jnc     $               ;Espera a que termine la conversion
        clr     EX0             ;Desabilita la interrupcion

        setb    out_ena         ;Habilita la salida de datos del conversor
        nop                     ;Genera un pulso
        mov     A,datos_conv    ;Lee el valor del conversor
        clr     out_ena         ;Termina el pulso
;
;*****************************************************************************
;
;     Termina la primera lectura e inicia la segunda
;

        setb    start
        nop
        clr     start

        setb    EX0
        clr     C
        jnc     $
        clr     EX0

        setb    out_ena
        nop
     mov   A,datos_conv
        clr     out_ena         ;Termina el pulso
        mov     dato_analogo,A  ;Guarda el dato en registro

        mov     B,dato_analogo  ;Rutina para convertir valor hexadecimal
                                ;en decimal de 3 digitos
     jz    sigue_dos            ;Si el dato leido es 0 termina la rutina
        clr     C               ;Borra registros que inciden en la
        clr     AC              ;conversion hexadecimal a bcd
     clr   A
```

```
        mov     cien,#0         ;Encera registro de almacenamiento de
                                ;mas de 100 unidades
;
;*****************************************************************************
;
;     Como el valor maximo a leer en el conversor es de 5 voltios con una
;     resolucion de 255 bits se debera sumar 2 numeros decimales por cada
;     bit del conversor
;

sumar_uno:

        add     A,#2            ;La rutina debe sumar 2 al acumulador
        da      A               ;hacer el ajuste decimal hasta que llegue
      jnc   sigue_uno           ;a 99 si pasa de este valor debe aumentar
        inc     cien            ;en 1 al registro de centenas
      clr   C
      clr   AC

sigue_uno:

      djnz  B,sumar_uno

sigue_dos:

        mov     B,A
        mov     R0,#(inicio_d+13)
        anl     A,#0FH
        orl     A,#30H
        mov     @R0,A
      mov   A,B
      swap  A
      mov   R0,#(inicio_d+12)
        anl     A,#0FH
        orl     A,#30H
      mov   @R0,A
      mov   R0,#(inicio_d+11)
      mov   @R0,#','
      mov   R0,#(inicio_d+10)
        mov     A,cien
      orl   A,#30H
      mov   @R0,A

        lcall     display
        lcall   del_lseg

        ret


;
;*****************************************************************************
;

fin_conversion:

        setb    C               ;Subrutina de atencion a la interrupcion
                                ;que genera el conversor cuando termina
      jnb   end_of_con,$        ;la conversion y tiene los datos de la
        reti                    ;ultima lectura listos para enviar
```

```
;
;*******************************************************************************
;

teclado:

      push  ACC
      push  PSW
      mov     A,P2                    ;Trae valor de la entrada de teclado
      anl     A,#0FH                  ;deja solo datos de teclado
        cjne    A,#0H,tecla_2

tecla_1:

      mov     dato_tecla,#uno
      ljmp    fin_tecla

tecla_2:

        cjne    A,#1H,tecla_3
      mov     dato_tecla,#dos
      ljmp  fin_tecla

tecla_3:

        cjne    A,#2H,tecla_4
      mov     dato_tecla,#sube
      ljmp  fin_tecla

tecla_4:

        cjne    A,#3H,tecla_5
      mov     dato_tecla,#tres
      ljmp  fin_tecla

tecla_5:

        cjne    A,#4H,tecla_6
      mov     dato_tecla,#cuatro
      ljmp  fin_tecla

tecla_6:

        cjne    A,#5H,tecla_7
      mov     dato_tecla,#cinco
      ljmp  fin_tecla

tecla_7:

        cjne    A,#6H,tecla_8
      mov     dato_tecla,#baja
      ljmp  fin_tecla

tecla_8:

        cjne    A,#7H,tecla_9
      mov     dato_tecla,#seis
      ljmp  fin_tecla
```

```
tecla_9:

        cjne    A,#8H,tecla_10
      mov     dato_tecla,#siete
      ljmp  fin_tecla

tecla_10:

        cjne    A,#9H,tecla_11
      mov     dato_tecla,#ocho
      ljmp  fin_tecla

tecla_11:

        cjne    A,#0AH,tecla_12
      mov     dato_tecla,#s2nd
      ljmp  fin_tecla

tecla_12:

        cjne    A,#0BH,tecla_13
      mov     dato_tecla,#nueve
      ljmp  fin_tecla

tecla_13:

        cjne    A,#0CH,tecla_14
      mov     dato_tecla,#clear
      ljmp  fin_tecla

tecla_14:

        cjne    A,#0DH,tecla_15
      mov     dato_tecla,#cero
      ljmp  fin_tecla

tecla_15:

        cjne    A,#0EH,tecla_16
      mov     dato_tecla,#enter
      ljmp  fin_tecla
tecla_16:

      mov     dato_tecla,#help
      ljmp  fin_tecla

fin_tecla:

        jnb     tecla_int,$             ;Espera al fin de la interrupcion
      pop   PSW
      pop   ACC
        mov     R7,#1                   ;Reduce el tiempo de espera al
        mov     hay_tecla,#'S'          ;al  minimo
      reti

;
;
;*****************************************************************************
```

```
;
;      Rutina de inicializacion temporizada del display
;

inicio_display:

      lcall del_20mili

        mov     port_dis,#0H            ;Apaga todas las salidas
      setb    ena
      orl     port_dis,#3H
      clr     ena

        lcall   del_10mili

        setb    ena
      orl     port_dis,#3H
      clr     ena

        lcall   del_700micro

        setb    ena
      orl     port_dis,#3H
      clr     ena

        lcall   del_700micro

        setb      ena
        anl     port_dis,#0FEH          ;Define 4 bits de interface
      clr   ena

        lcall   del_700micro

;*****************************************************************************
;
;

        lcall   del_700micro

        mov     data_dis,#28H           ;Define dos lineas y 5x7 dots

        lcall   saca_display
     lcall   clrD
        lcall   del_700micro

        mov     data_dis,#0AH           ;SET display OFF CURSOR ON

        lcall   saca_display
     lcall   clrD

        mov     data_dis,#0EH           ;SET display AND CURSOR ON

        lcall   saca_display
     lcall   clrD

        mov     data_dis,#06H           ;SET INC add 1,SHIFT CUR RIGHT

        lcall   saca_display
        lcall   borrar_display                  ;Borra el display
```

```
        lcall   del_700micro

        ret

;
;*****************************************************************************
;

borrar_display:

        lcall   clrd                    ;Rutina para borrar el display

        setb      ena
            mov         data_dis,#01H

        lcall   saca_display

        ret


;
;*****************************************************************************
;

saca_display:

        lcall   del_700micro        ;Rutina para sacar datos que vienen en

        anl     port_dis,#0F0H      ;data_dis al area de display
        mov     A,data_dis          ;Como estamos trabajando con una
        swap    A                   ;interfase
        anl     A,#0FH              ;de 4 bits se debe mandar: primero los
        orl     port_dis,A          ;4 nibles menos significativos y luego

        setb    ena                 ;los mas significativos
      clr   r_w

        lcall   del_500micro

            clr         ena

        lcall   del_500micro

      mov   A,data_dis
      anl   A,#0FH
      anl   port_dis,#0F0H
      orl   port_dis,A
      setb  ena

        lcall   del_500micro

            clr         ena

        ret


;
;*****************************************************************************
;
```

```
clrd:   clr     rs                  ;Baja las seniales de control del display
        clr     r_w
        clr     ena

        ret

;
;*******************************************************************************
;
;     Area de definicion de letreros fijos para el display
;


regis_disp:

      mov     R1,#(0*16)
      ljmp    sacle

conversor_dis:

      mov     R1,#(1*16)
      ljmp    sacle

falla_dis:

      mov     R1,#(2*16)
      ljmp    sacle

SACL4:

      mov     R1,#(3*16)
      ljmp    SACLE

SACL5:

      mov     R1,#(4*16)
      ljmp    SACLE

SACL6:

      mov     R1,#(5*16)
      ljmp    SACLE

SACL7:

      mov     R1,#(6*16)
      ljmp    SACLE

SACL8:
      mov     R1,#(7*16)
      ljmp    SACLE


SACL9:

      mov     R1,#(8*16)
      ljmp    SACLE
```

```
SACL10:

      mov     R1,#(9*16)
      ljmp    SACLE

SAC2000:

      mov     R1,#(10*16)
      ljmp    SACLE


opcion_1_dis:

        mov     R1,#(11*16)
      ljmp    SACLE

magnitudes_dis:

      mov     R1,#(12*16)
      ljmp    SACLE

intervalo_dis:

      mov     R1,#(13*16)
      ljmp    SACLE

solo_tem_dis:

      mov     R1,#(14*16)
      ljmp    SACLE

;
;*******************************************************************************
;


sacle:

        mov     A,R1                    ;Direccion del letrero
        add     A,#09H                  ;Constante del Program counter
        mov     R1,A
        mov     R0,#inicio_d            ;Direccion inicial del buffer de
                                        ;display
11:
        mov     A,R1                    ;Pone la direccion del caracter
        movc    A,@A+PC                 ;Lee el caracter de la tabla
        mov     @R0,A                   ;Lo guarda en el buffer del display
      inc     R0
      inc     R1
      mov     A,R0
      xrl     A,#(inicio_d+16)          ;En total debe leer 16 caracteres
      jnz     11

      ret

;
;*******************************************************************************
;
```

```
display_00:DB 'TRABAJANDO. OK. '
display_01:DB 'Conv:           V'
display_02:DB '*Tecla invalida*'
display_03:DB '-Control  reloj-'
display_04:DB 'Minutos    00-59'
display_05:DB 'Hora       00-23'
display_06:DB 'Dia Semana   0-7'
display_07:DB 'Fecha      01-31'
display_08:DB 'Mes        01-12'
display_09:DB 'Anio dece  00-99'
display_10:DB 'Anio mil   19-20'
display_11:DB 'Interva. ** Min.'
display_12:DB '1>T 2>H 3>M  4>F'
display_13:DB 'Interva: ** Min.'
display_14:DB 'Temper:       ßC'



;
;*******************************************************************************
;


solo_hum_dis:

      mov     R1,#(0*16)
      ljmp    SACLE1

solo_mon_dis:

      mov     R1,#(1*16)
      ljmp    SACLE1

no_registro_dis:

      mov     R1,#(2*16)
      ljmp    SACLE1

conversor_nro_dis:

      mov     R1,#(3*16)
      ljmp    SACLE1

memoria_full:

      mov     R1,#(4*16)
      ljmp    SACLE1

epn_disp:

        mov     R1,#(5*16)
      ljmp    SACLE1

titulo2:
        mov     R1,#(6*16)
      ljmp    SACLE1
```

```
titulo3:
        mov     R1,#(7*16)
      ljmp    SACLE1

titulo4:
        mov     R1,#(8*16)
      ljmp    SACLE1

titulo5:
        mov     R1,#(9*16)
      ljmp    SACLE1

titulo6:

        mov     R1,#(10*16)
      ljmp    SACLE1

titulo7:
        mov     R1,#(11*16)
      ljmp    SACLE1




;
;*****************************************************************************
;


saclel:

        mov     A,R1
        add     A,#09H
      mov     R1,A
        mov     R0,#inicio_d

111:

        mov     A,R1
        movc    A,@A+PC
        mov     @R0,A
      inc     R0
      inc     R1
      mov     A,R0
      xrl     A,#(inicio_d+16)
      jnz     111

      ret



;
;*****************************************************************************
;

display_15:DB 'Humed.:        %'
display_16:DB 'Monoxi:      PPM'
```

```
display_17:DB '** NO registra *'
display_18:DB 'Nro. de entrada '
display_19:DB 'Memoria *llena* '

;
;*****************************************************************************
;

display_20:DB '   E.  P.  N.    '
display_21:DB 'ING. ELECTRICA.  '
display_22:DB 'TESIS DE GRADO.  '
display_23:DB '   AUSPICIO.     '
display_24:DB '  P-BID-081.     '
display_25:DB '  E. M. C. A.    '
display_26:DB 'M.CHISAGUANO.A.  '


display:

        lcall   clrd

        mov     data_dis,#80H           ;Pone la direccion 00 del display

        lcall   saca_display

        mov     R0,#inicio_d            ;ubica la direccion de inicio del
                                        ;display
cont_espe:

        mov     R6,#num_car             ;Define la cantidad de caracteres
                                        ;que puede manejar en esta area
cambio:
        lcall   alterno                 ;Llama a rutina que manda el caracter
        inc     R0                      ;al display
        djnz    R6,cambio

segunda_mitad:

        lcall   clrd
        mov     data_dis,#0C0H          ;Pone la direccion 40h del display
        lcall   saca_display
      mov     R6,#num_car

cambio1:

        lcall   alterno
      inc     R0
        djnz    R6,cambio1

sale_disp:

        clr     C                       ;Borra el carry para control de salida
      ret

;
;*****************************************************************************
;
```

```
alterno:

      setb    rs              ;Realiza el control del protocolo
      clr     r_w             ;de comunicacion con el display
      clr     ena             ;Cargando el valor del ACC en el display
      setb    ena
      mov     A,@R0
      mov     data_dis,A
      lcall   saca_display
      clr     rs
      ret

;
;*****************************************************************************
;

del_1seg:

      mov   R7,#5
      mov     TH0,#60H
      mov     TL0,#4AH
      setb    TR0

lazo_1seg:

      jnb   TF0,$
      clr   TF0
      djnz  R7,lazo_1seg
      clr   TR0
      ret

;
;*****************************************************************************
;

del_5seg:

      mov     R6,#5                 ;Hace el lazo por 5 segundos esperando

lazo_del5:                          ;que se presione otra tecla, si no se

      lcall   del_1seg              ;presiona sale por timeout
      mov   A,hay_tecla
      cjne  A,#'S',sigue_del
      ret

sigue_del:

      djnz  R6,lazo_del5
      ret

;
;*****************************************************************************
;

del_20mili:

      mov     TH0,#0E8H
      mov     TL0,#54H
```

```
        setb  TR0
        jnb   TF0,$
        clr   TF0
        clr   TR0
        ret

;
;*****************************************************************************
;

del_10mili:

          mov     TH0,#0F4H
          mov     TL0,#2AH
        setb  TR0
        jnb   TF0,$
        clr   TF0
        clr   TR0
        ret

;
;*****************************************************************************
;

del_700micro:

        mov   TH0,#0FFH
          mov     TL0,#2CH
        setb  TR0
        jnb   TF0,$
        clr   TR0
        clr   TF0
        clr   TR0
        ret

;
;*****************************************************************************
;

del_500micro:

        mov   TH0,#0FFH
          mov     TL0,#069H
        setb  TR0
        jnb   TF0,$
        clr   TR0
        clr   TF0
        clr   TR0
        ret
;
;
;*****************************************************************************
;
;       Rutina para leer el reloj y ponerlo en el display
;
reloj:

        lcall   leer_reloj                ;Pone en condicion de leer
```

```
        mov     R0,#inicio_d
        mov     A,diam_dat

        lcall   mu_ascii_display

        mov     @R0,#'-'
        inc     R0
        mov     A,mese_dat

        lcall   mu_ascii_display

        mov     @R0,#'-'

      inc     R0

      mov   DPTR,#anio_2000
      movx  A,@DPTR

        lcall   mu_ascii_display

        mov     A,anio_dat

        lcall   mu_ascii_display

        mov     @R0,#' '
        inc     R0
        mov     @R0,#' '
        mov     A,hora_dat

        lcall   mu_ascii_display

        mov     @R0,#':'
        inc     R0
        mov     A,min_dat          ;Muestra en display "ANIO-MES-DIA HORA:MIN"

        lcall   mu_ascii_display

        mov     @R0,#' '
      ret


;
;*****************************************************************************
;

leer_reloj:

        lcall   secuencia_reloj            ;Pone en condicion de leer

next_lee:

        lcall   lee_reg

        lcall   lee_reg
        mov     seg_dat,A
        lcall   lee_reg
      mov     min_dat,A
        lcall   lee_reg
      mov     hora_dat,A
        lcall   lee_reg
      mov     dias_dat,A
```

```
        lcall   lee_reg
      mov     diam_dat,A
        lcall   lee_reg
      mov     mese_dat,A
        lcall   lee_reg
      mov     anio_dat,A
      ret


;
;*****************************************************************************
;
;mu_ascii_display       rutina para convertir en ascci valor hexa y guardar
;                       en posiciones sucesivas de ram para display
;


mu_ascii_display:

      mov     B,A
      swap    A
      anl     A,#0FH
      add     A,#30H                  ;display requiere datos en ASCII
      mov     @R0,A
      inc     R0
      mov     A,B
      anl     A,#0FH
      add     A,#30H
      mov     @R0,A
      inc     R0
      ret


;
;*****************************************************************************
;

iguala:

      lcall   display
      clr     C
      jnc     $                       ;Espera por tecla
      clr     C
      mov     A,B
      anl     A,#0FH
      swap    A
        mov     espera,A              ;Guarda primer bcd
      clr     C
        jnc     $                     ;Espera por tecla
      clr     C
      mov     A,B
      anl     A,#0FH
      add     A,espera
        lcall   gra_reg
      ret


;
;*****************************************************************************
```

```
;
;
;secuencia_reloj
;
;       rutina para ejecutar la secuencia de lecturas y escrituras
;       necesarias para establecer la comunicacion con el reloj.
;       La subrutina regresa dejando abierto el reloj para mantener
;       la comunicacion, el ACC y el registro B quedan modificados.
;

secuencia_reloj:

        lcall   cerrar_reloj            ;Asegura que el reloj esta cerrado
        mov     B,#4
        mov     A,#0C5H

secuencia_relojA:

        lcall   gra_reg
        xrl     A,#0FFH
        lcall   gra_reg
        swap    A
        djnz    B,secuencia_relojA
        ret

;
;*****************************************************************************
;
;       cerrar_reloj  rutina para asegurar que todos los registros del reloj
;       estan cerrados.
;

cerrar_reloj:

        mov     B,#9

loop_cer:

        lcall   lee_reg
      djnz    B,loop_cer
      ret

;
;*****************************************************************************
;
;leereg rutina para leer datos del reloj , devuelve el valor leido
;en el ACC
;

lee_reg:

      push    DPL             ;guarda el DATA POINTER en el stack
      push    DPH
      push    MCON
        orl     MCON,#4
        push    B
        mov     DPL,#4
      mov     DPH,#0
        mov     B,#8
```

```
loop_lee:

        push    ACC
        movx    A,@DPTR
        rlc     A
        pop     ACC
        rrc     A
        djnz    B,loop_lee
        pop     B
        pop     MCON
        pop     DPH
        pop     DPL
        ret


;
;*****************************************************************************
;
;gra_reg Rutina para grabar un registro desde el ACC en el reloj
;El ACC trae valor a guardar
;


gra_reg:

        push    DPL
        push    DPH
        push    MCON
        ORL     MCON,#4
        push    B
        mov     DPH,#0
        mov     B,#8

loop_gra:

        push    ACC
        anl     A,#1
        mov     DPL,A
        movx    @DPTR,A
        pop     ACC
        rr      A
        djnz    B,loop_gra
        pop     B
        pop     MCON
        pop     DPH
        pop     DPL
        ret



;
;*****************************************************************************
;
;       Dato recibido desde el computador
;

com_serial:
```

```
        push  ACC
        push  B
        mov     A,SBUF
        mov     dato_leido,A
        lcall datos_cpu
        pop   B
        pop   ACC
        reti

;
;*******************************************************************************
;
; datos_cpu
;                   Rutina para tomar datos desde el  computador.
;             La cpu debe mandar un header luego datos y fin de archivo
;             El header puede ser 'I' para igualar el reloj, 'Z' para
;
;                   Para igualar reloj el cpu debe mandar la informacion de
;             fecha y hora asi:
;
;                   I MM-DD-AAAA HH:MM:SS


datos_cpu:

        clr   RI
        mov   A,dato_leido
        mov   B,A
        cjne    A,#'I',com_serial_datol
        ljmp    igualar                         ;Iguala el reloj

;
;*******************************************************************************
;
com_serial_datol:

          cjne    A,#'Z',com_serial_dato2
          ljmp    vaciar_mem                    ;Encerar registros

;
;*******************************************************************************
;

com_serial_dato2:

          cjne    A,#'L',com_serial_dato3
          ljmp    leer                          ;Mandar datos al cpu

;
;*******************************************************************************
;




com_serial_dato3:
```

```
        clr   RI
        clr   TI
        ret

;
;*****************************************************************************
;
;       Rutina para mandar datos al computador.
;       Esta rutina envia todos los datos almacenados desde el ultimo
;       encerado.
;


leer:

        push  DPL
        push  DPH
        push  ACC
        push  PSW
        mov   DPTR,#datos
          mov     A,#'L'          ;Devuelve caracter para indicar que lo
          lcall   send_car        ;recibio
        mov   a,#car_ret
        lcall send_car

lazo_mandar:

          movx      A,@DPTR
        lcall send_car             ;Manda el tipo de registro:
                                   ;T => temperatura, H => humedad
                                   ;M => monoxido
          mov     A,#','           ;Cada dato va separado por coma para que
        lcall send_car             ;pueda ser manejado en la hoja electronica

        inc   DPTR
        movx  A,@DPTR

          cjne    A,#99H,Dos_mil     ;Verificamos si es el anio 99
        mov   A,#19H
        lcall send_dat
        ljmp  Noventa
Dos_mil:
        mov   A,#20H
        lcall send_dat

Noventa:
          movx    A,@DPTR            ;Recuperamos el anio
          lcall       send_dat       ;Manda los dos caracteres del calendrio
        mov   A,#','
        lcall send_car

        inc   DPTR
        movx  A,@DPTR
        lcall send_dat
        mov   A,#','
        lcall send_car

        inc   DPTR
        movx  A,@DPTR
```

```
        lcall send_dat
        mov   A,#','
        lcall send_car

        inc   DPTR
        movx  A,@DPTR
        lcall send_dat
          mov     A,#':'
        lcall send_car

        inc   DPTR
        movx  A,@DPTR
        lcall send_dat
        mov     A,#','
        lcall send_car

        inc   DPTR
        movx  A,@DPTR


         lcall   send_car


        mov     A,#car_ret
        lcall send_car

          inc     DPTR                    ;Hace el lazo hasta que se terminen
          movx    A,@DPTR                 ;los datos
        cjne  A,#'Z',lazo_mandar

          mov     A,#26                   ;Para finalizar el envio de todos
          lcall   send_car                ;los registros manda un cntrl_z

        mov   A,#car_ret
        lcall send_car

        pop   PSW
        pop   ACC
        pop   DPH
        pop   DPL
        ret

;
;*******************************************************************************
;

send_dat:

        mov     B,A             ;Guarda temporal
        swap    A
        anl     A,#0FH          ;Lo transforma en ASCII
        orl   A,#30H
        clr   TI

        lcall manda_rs

          mov     A,B             ;Recupera el datos original
          anl     A,#0FH          ;Lo transforma en ASCII
        orl   A,#30H
```

```
      lcall manda_rs
      ret

;
;*****************************************************************************
;

manda_rs:

      mov   SBUF,A
      jnb     TI,$              ;Espera al fin de transmision
      clr   TI
      ret

;
;*****************************************************************************
;

send_car:

      clr   TI
      lcall manda_rs
      ret

;
;*****************************************************************************
;
;       Rutina para encerar los registros de almacenamiento en la NVRAM
;
vaciar_mem:

      mov   DPTR,#ultimo_reg
      mov     A,#0H
      movx  @DPTR,A
      mov   DPTR,#minutos_r
      mov     A,#0H                   ;Ultimo registro de minutos = 0
      movx  @DPTR,A
      mov     DPTR,#datos             ;Inicia los punteros en el inicio
      mov   R0,#10
      mov   A,#'Z'

lazo_vaciar_mem:

      movx  @DPTR,A
      inc   DPTR
      djnz    R0,lazo_vaciar_mem
      mov   DPTR,#datos
      mov   A,DPL
      mov   B,A
      mov   A,DPH
      mov   DPTR,#direcc_hig
      movx  @DPTR,A
      mov   A,B
      mov   DPTR,#direcc_low
      movx  @DPTR,A
      ret                             ;de la tabla de datos
;
;*****************************************************************************
;
```

```
igualar:

      mov   A,#'I'          ;Devuelve caracter para indicar que lo
      lcall send_car        ;recibio
      mov   a,#car_ret
      lcall send_car


imes:

      lcall caracter
      mov   mese_dat,A
      jnb   RI,$
        clr     RI

diames:

      lcall caracter
      mov   diam_dat,A
      jnb   RI,$
        clr     RI


dece_anios:

      lcall caracter
      push  DPL
      push  DPH
      mov   DPTR,#anio_2000
      movx  @DPTR,A
      pop   DPH
      pop   DPL

anios:

      lcall caracter
      mov   anio_dat,A

horas:

      lcall caracter
      mov   hora_dat,A
      jnb   RI,$
      clr     RI
minutos:

      lcall   caracter
      mov   min_dat,A
      jnb   RI,$
      clr     RI

segundos:

      lcall caracter
      mov   seg_dat,A

;
;*******************************************************************************
```

```
;
load_clk:

        lcall   secuencia_reloj
        mov     A,#0            ;Pone en reloj .00 segundos
        lcall   gra_reg
      mov   A,seg_dat
        lcall   gra_reg
      mov   A,min_dat
        lcall   gra_reg
      mov   A,hora_dat
        lcall   gra_reg
        mov     A,#01H          ;Pone en reloj el primer dia
        lcall   gra_reg
      mov   A,diam_dat
        lcall   gra_reg
      mov   A,mese_dat
        lcall   gra_reg
      mov   A,anio_dat
        lcall   gra_reg
      clr   RI
        ret                     ;Termina rutina de igualacion

;
;*****************************************************************************
;

caracter:

        jnb     RI,$            ;Espera proximo caracter
      clr   RI
        mov     A,SBUF          ;Lee caracter en ASCII
        anl     A,#0FH          ;Lo transforma en HEXA
      swap  A
        mov     B,A             ;Guarda dato
        jnb     RI,$            ;Espera proximo caracter
      clr   RI
        mov     A,SBUF
        anl     A,#0FH
      add   A,B
      ret

;
;
;*****************************************************************************
;
;       El valor de la lectura de monoxido esta en 'dato_analogo'
;

valor_mono:

        mov     A,dato_analogo
      mov   DPTR,#tabla_mono

        jz      salir_mono

lazo_mono:
```

```
        inc     DPTR
        inc     DPTR

      dec   A
      jnz   lazo_mono

salir_mono:

        movc    A,@A+DPTR
        mov     dato_mono1,A
      inc   DPTR
      clr   A
        movc    A,@A+DPTR
        mov     dato_mono2,A
        ret
;
;
;*******************************************************************************
;

tabla_mono:


DB    22H,57H,22H,12H,21H,69H,21H,26H,20H,83H,20H,42H,20H,02H,19H,62H
DB    19H,23H,18H,85H,18H,48H,18H,11H,17H,75H,17H,40H,17H,06H,16H,72H
DB    16H,39H,16H,06H,15H,75H,15H,43H,15H,13H,14H,83H,14H,54H,14H,25H
DB    13H,97H,13H,69H,13H,42H,13H,15H,12H,89H,12H,64H,12H,39H,12H,14H
DB    11H,90H,11H,67H,11H,43H,11H,21H,10H,99H,10H,77H,10H,56H,10H,35H
DB    10H,14H,9H,94H,9H,74H,9H,55H,9H,36H,9H,18H,8H,99H,8H,82H
DB    8H,64H,8H,47H,8H,30H,8H,14H,7H,98H,7H,82H,7H,66H,7H,51H
DB    7H,36H,7H,22H,7H,08H,6H,94H,6H,80H,6H,66H,6H,53H,6H,40H
DB    6H,28H,6H,15H,6H,03H,5H,91H,5H,79H,5H,68H,5H,57H,5H,46H
DB    5H,35H,5H,24H,5H,14H,5H,04H,4H,94H,4H,84H,4H,74H,4H,65H
DB    4H,56H,4H,47H,4H,38H,4H,29H,4H,21H,4H,12H,4H,04H,3H,96H
DB    3H,88H,3H,81H,3H,73H,3H,66H,3H,58H,3H,51H,3H,44H,3H,38H
DB    3H,31H,3H,24H,3H,18H,3H,12H,3H,05H,2H,99H,2H,93H,2H,88H
DB    2H,82H,2H,76H,2H,71H,2H,66H,2H,60H,2H,55H,2H,50H,2H,45H
DB    2H,40H,2H,36H,2H,31H,2H,26H,2H,22H,2H,17H,2H,13H,2H,09H
DB    2H,05H,2H,01H,1H,97H,1H,93H,1H,89H,1H,85H,1H,82H,1H,78H
DB    1H,74H,1H,71H,1H,68H,1H,64H,1H,61H,1H,58H,1H,55H,1H,52H
DB    1H,49H,1H,46H,1H,43H,1H,40H,1H,37H,1H,35H,1H,32H,1H,29H
DB    1H,27H,1H,24H,1H,22H,1H,19H,1H,17H,1H,15H,1H,12H,1H,10H
DB    1H,08H,1H,06H,1H,04H,1H,02H,1H,00H,0H,98H,0H,96H,0H,94H
DB    0H,92H,0H,90H,0H,88H,0H,87H,0H,85H,0H,83H,0H,82H,0H,80H
DB    0H,78H,0H,77H,0H,75H,0H,74H,0H,72H,0H,71H,0H,70H,0H,68H
DB    0H,67H,0H,65H,0H,64H,0H,63H,0H,62H,0H,60H,0H,59H,0H,58H
DB    0H,57H,0H,56H,0H,55H,0H,54H,0H,53H,0H,52H,0H,50H,0H,49H
DB    0H,49H,0H,48H,0H,47H,0H,46H,0H,45H,0H,44H,0H,43H,0H,42H
DB    0H,41H,0H,41H,0H,40H,0H,39H,0H,38H,0H,37H,0H,37H,0H,36H
DB    0H,35H,0H,35H,0H,34H,0H,33H,0H,33H,0H,32H,0H,31H,0H,31H
DB    0H,30H,0H,29H,0H,29H,0H,28H,0H,28H,0H,27H,0H,27H,0H,26H
DB    0H,26H,0H,25H,0H,25H,0H,24H,0H,24H,0H,23H,0H,23H,0H,22H
DB    0H,22H,0H,21H,0H,21H,0H,21H,0H,20H,0H,20H,0H,19H,0H,19H
DB    0H,19H,0H,18H,0H,18H,0H,17H,0H,17H,0H,17H,0H,16H,0H,16H
DB    0H,16H,0H,16H,0H,15H,0H,15H,0H,15H,0H,14H,0H,14H,0H,14H
DB    0H,13H,0H,13H,0H,13H,0H,13H,0H,12H,0H,12H,0H,12H,0H,12H
DB      0H,11H,0H,11H,0H,11H,0H,11H,0H,11H,0H,10H,0H,10H,0H,10H
DB      0H,10H,0H,10H,0H,09H,0H,09H,0H,09H,0H,09H,0H,09H,0H,09H
DB      0H,08H,0H,08H,0H,8H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H
```

```
DB      0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H
DB      0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H,0H


;
;
;*****************************************************************************
;
;       El valor de la lectura de temperatura esta en 'dato_analogo'
;

valor_tempera:

        mov     B,dato_analogo

        mov     A,B
      mov   cien,#0
      jz    sigue_dosa
      clr   C
      clr   AC
      clr   A

sumar_unoa:

        add     A,#2            ;Puesto que la maxima lectura equivale
        da      A               ;a 50.0 grados osea 500 posiciones, pero
      jnc   sigue_unoa          ;el conversor entrega hasta 255 bits sera
        inc     cien            ;necesario sumar 2 unidades decimales por
        clr     C               ;cada bit de lectura
      clr   AC

sigue_unoa:

      djnz  B,sumar_unoa

sigue_dosa:

      mov   dato_temp2,A
      mov   A,cien
      mov   dato_temp1,A
      ret

;
;
;*****************************************************************************
;
;       El valor de la lectura de humedad esta en 'dato_analogo'
;

valor_humedad:

        mov     B,dato_analogo
        mov     A,B
      mov   cien,#0
      jz    sigue_dosb
      clr   C
      clr   AC
      clr   A
```

```
sumar_unob:

      add     A,#4            ;Puesto que la maxima lectura equivale
      da      A               ;a 100.0 % osea 1000 posiciones, pero
    jnc   sigue_unob          ;el conversor entrega hasta 255 bits sera
      inc     cien            ;necesario sumar 4 unidades decimales por
      clr     C               ;cada bit de lectura
    clr   AC

sigue_unob:

      djnz  B,sumar_unob

sigue_dosb:

      mov   dato_hume2,A
      mov   A,cien
      mov   dato_hume1,A
      ret

;
;
;*****************************************************************************
;
;     Origen del area NVRAM
;

      org   1800H

intervalo         equ   1800H
inter_dec         equ   1801H
minutos_r         equ   1802H
registra          equ   1803H
ultimo_reg        equ   1804H
direcc_low        equ   1805H
direcc_hig        equ   1806H
anio_2000         equ   1807H

;
;
;*****************************************************************************
;

      org   1810H


datos:









;
;
```

```
;*****************************************************************************************
;
;*****************************
;       ******************
;            *******
;               *
;
        END
;
;               *
;            *******
;       ******************
;*****************************
```

MANUAL DEL USUARIO

E.M.C.A.

CONTENIDO

Páginas

MANUAL DEL USUARIO

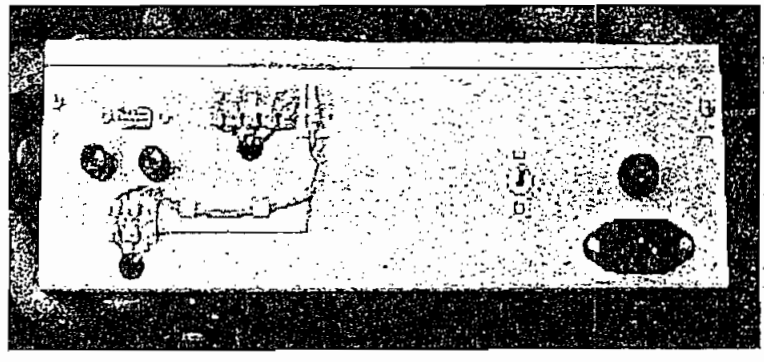
1 Descripción del equipo.....3
2 Características técnicas.....4
3 Condiciones de operación del equipo.....5
4 Manejo del equipo.....6
Valor del registro.....7
Intervalo.....7
5 Mantenimiento del equipo.....8
Circuito del sensor de temperatura.....8
Ajuste a cero.....9
Calibración de la ganancia.....9
Circuito del sensor de humedad.....9
6 Utilización del programa de procesamiento en el computador.....10

MANUAL DEL USUARIO.

El equipo esta diseñado para poder trabajar de una manera eficiente y sin complicaciones, permitiendo de esta forma que el usuario pueda utilizar al máximo su capacidad y aprovechar sus resultados.

Este manual ayudará a familiarizarse con el EMCA (Equipo de Monitoreo de Contaminación del Aire) y sus características. Describe su modo de uso, así como también el programa de procesamiento que se lo corre desde un PC.

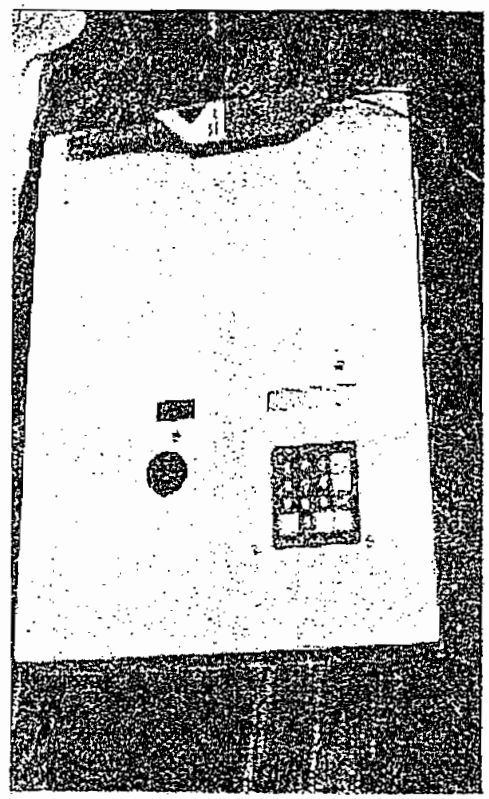
1. DESCRIPCION DEL EQUIPO.



A) VISTA POSTERIOR

En la figura A se tiene la vista posterior donde se puede apreciar:

- Un conector DB–9 que permite la comunicación serial entre el equipo y un computador.
- Puntos de conexión para el sensor de humedad con polaridades definidas; esto es, rojo (+) y negro (-).
- Cinco puertos disponibles para incrementar más sensores a futuro.
- El sensor de temperatura (LM335).
- Conmutador de energización del equipo.
- Portafusible, que contiene un fusible de 2 Amp..
- Conector polarizado, para el cable de energía.



B) VISTA FRONTAL

En la figura B se tiene la vista frontal donde se puede apreciar:

- El sensor de humedad.
- El sensor de monóxido de carbono con su display incorporado.
- Un display LCD de una línea y 16 caracteres por fila, en donde se indican los valores de: temperatura, humedad, monóxido de carbono, fecha y hora.
- Teclado de 16 teclas.

2. CARACTERISTICAS TECNICAS

1. Voltaje de entrada (rms)	100 -240 ± 10 % Vac, 3.2 A,
2. Frecuencia de trabajo	47 - 63 Hz
3. Protección (fusible)	250 V / 3.5 A
4. Voltaje de alimentación a los circuitos	
• Vcc	5 Vdc, 12 A

• Vee			+ 12 Vdc,		3 A
• Vss			- 12 Vdc,		1 A
5. Temperatura					
• Rango de operación		0	a	50	°C
• Rango de medición		0	a	50	°C
6. Humedad*					
• Rango de operación(no condensado)		0	a	96	% RH
• Rango de medición		0	a	96	% RH
7. Monóxido de Carbono**					
• Rango de operación		0	a	2257	ppm
• Rango de medición		12	a	1000	ppm
8. Dimensiones	Aprox.	300x100x400 mm (an/al/prf)			
9. Peso					4.0 Kg
10. Accesorios suministrados					
• Cable de alimentación de CA					1
• Cable de transmisión de datos con switch y conectores DB9					1
11. Comunicación serial					
• Velocidad					2400 bps
• Longitud de la palabra					8 bits
• Bit de parada					1 bit
• Paridad					ninguna
12. Puertos opcionales***					
• Puertos disponibles de entradas al conversor					5
13. Forma de trabajo					
• Solamente en forma horizontal y en lugares firmes.					

3 CONDICIONES DE OPERACIÓN DEL EQUIPO.

Se lo puede hacer funcionar en cualquier lugar, que sea requerido; sin embargo, se deben tomar en cuenta las siguientes recomendaciones a fin de obtener buenos resultados.

- Mantener siempre al equipo en un entorno limpio y seco. Asegúrese de que la superficie sobre la que reposa es lisa y firme.
- No coloque ningún objeto encima del equipo ni cubra el orificio de ventilación, este orificio permite absorbe el aire del medio ambiente para ser sensado y sirve también para refrigerar el equipo.

---

* Funciona con una fuente de voltaje independiente de 9 Vdc

** Dispone de un display que solamente presenta valores referenciales y promediados durante 2,5 minutos.

A continuación se limpia el display y aparece otro mensaje en forma repetitiva durante todo el tiempo que se encuentre tomando las respectivas mediciones, es el siguiente:

E.M.C.A.[1]
M. CHISAGUANO. A.
TRABAJANDO OK.
FECHA Y HORA
VALOR DEL REGISTRO
INTERVALO.

Donde:

Valor del registro significa que puede ser solamente: Temperatura (°C), Humedad (%RH), monóxido de Carbono (PPM) o todas.

Intervalo es el espacio de tiempo en el cual se van a almacenar en la memoria la variable o las variables descritas anteriormente y cuyo rango es de 01 minutos a 99 minutos.

Cuando se requiere cambiar el intervalo de tiempo o la toma de mediciones de cualquiera de las variables descritas, se debe proceder de la siguiente forma:

- Digite 2nd y espere a que se detenga el ciclo.
- Ponga la clave 123 <enter>[2].
- Elija el intervalo de tiempo en el cual se va ha almacenar los datos.
- Elija la variable con las siguientes opciones:
  1. Temperatura.
  2. Humedad.
  3. Monóxido de carbono.
  4. Todas.

Si por alguna razón durante este procedimiento no elige adecuadamente, éste termina y sigue operando sin alterar en nada la última programación ingresada.

[1] Equipo de Monitoreo de Contaminación del Aire.
[2] Si es <help> significa que puede elegirse ver si trabaja cualquiera de las entradas del conversor.

5 MANTENIMIENTO DEL EQUIPO.

El equipo requiere de un mantenimiento preventivo mensual, por cuanto su forma de operar, es en el campo, como consecuencia de ello se pueden acumular partículas de polvo y afines especialmente en los sensores. La forma de mantenimiento de los sensores instalados debe ser con aire seco, sin utilizar ningún tipo de químico.

Cuando se requiera calibrar las tarjetas de control de cada uno de los sensores instalados se deben seguir todos los siguientes procedimientos:

- Circuito del sensor de temperatura.

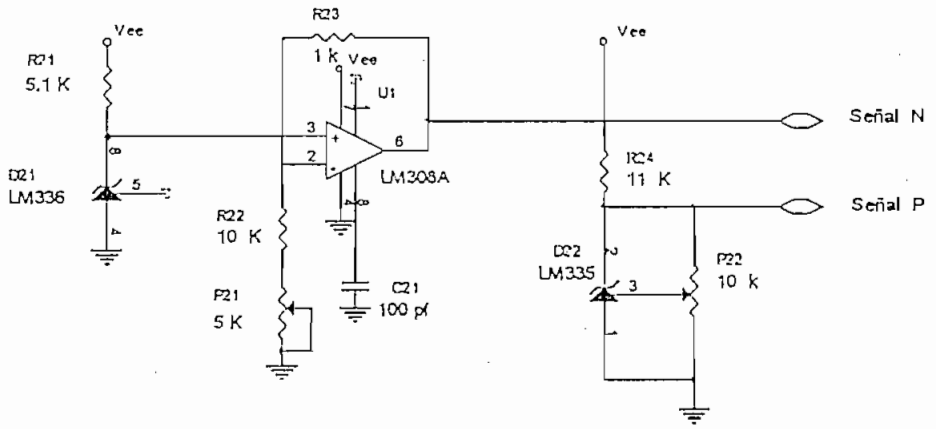


FIGURA 1 CIRCUITO DE REFERENCIA DE SEÑAL DE TEMPERATURA

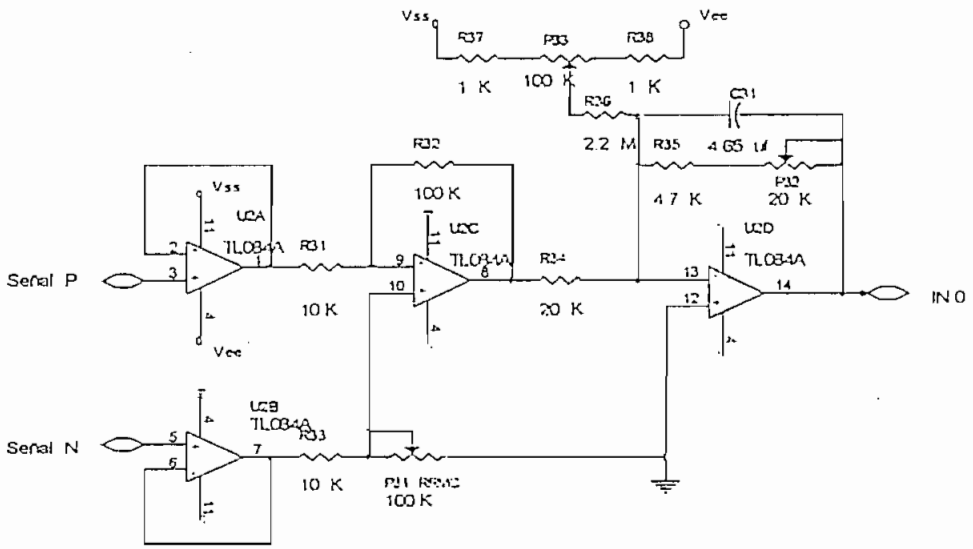


FIGURA 2 CIRCUITO AMPLIFICADOR DEL SENSOR DE TEMPERATURA

Ajuste a cero.

Para realizar la calibración del circuito de la figura 1 se debe medir el voltaje en el punto de señal N y ajustar el potenciometro P_{21} hasta conseguir exactamente 2.73 V. Luego como comparación también se debe ajustar el voltaje del sensor. LM335 a 2.73 V y comprobar que las señales N y P con respecto a una referencia común deben medir exactamente 2.73 V, finalmente se debe medir en el punto 1 de la figura 2 y ajustar el potenciometro P_{31} hasta encontrar un valor igual a 0 V.

Calibración de la ganancia.

Al igual que el ajuste a cero, es importante calibrar el potenciometro P_{21} hasta conseguir exactamente 3.23 V que viene ha ser el equivalente a 50 °C y su procedimiento se realiza con el potenciometro P33 y su control con el potenciometro P_{32} hasta llegar a medir en el punto 14 + 5V.

Al seguir todos estos pasos se conseguirá con seguridad, que el equipo este listo para operar en un rango de 0°C a 50 °C. No está por demás hacer mediciones en puntos intermedios.

- Circuito del sensor de humedad.

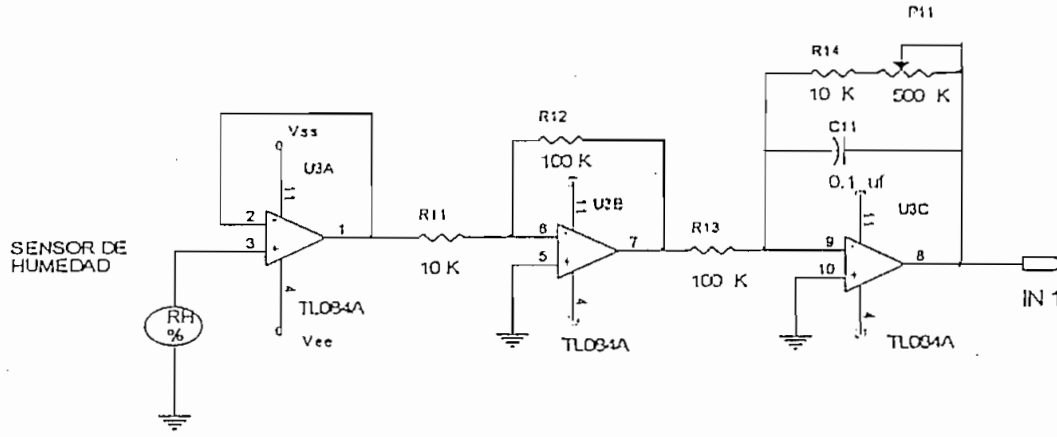


FIGURA 3 CIRCUITO AMPLIFICADOR DEL SENSOR DE HUMEDAD

Para calibrar el circuito de amplificación del sensor de humedad se debe ajustar el potenciometro P_{11} comparando con el voltaje de entrada y multiplicando por la ganancia

equivalente, se recomienda realizar este ajuste en un lugar seco y estable, preferible un ambiente cerrado.

6. - UTILIZACIÓN DEL PROGRAMA DE PROCESAMIENTO EN EL COMPUTADOR.

Al utilizar el programa de procesamiento en el computador este tiene, la finalidad de poder realizar la lectura de los datos almacenados en la memoria, igualar el reloj calendario y borrar los datos cuando se requiera.

El procedimiento se lo realiza al igual que cualquier programa de aplicación, utilizándose las mismas opciones que cualquier hoja electrónica; es decir, se pueden ver los gráficos, tablas e imprimir.

El programa se llama EMCA, y es un arreglo de diversas aplicaciones existentes en el mercado siendo de esta manera un complemento muy útil para el equipo.

Para poder utilizar el programa de procesamiento se requiere de un computador con las siguientes características:

- Procesador Pentium compatible.
- Velocidad del procesador 100 Mhz mínimo
- Memoria 32k en RAM
- Espacio en disco 10Mb mínimo
- Mouse
- Drive 3.5"
- Monitor SVGA color.
- Teclado
- Windows95.
- Nota: el programa EMCA es una aplicación de 32 bits, se debe actualizar en ODBC (medidas.mad). y sus claves de instalación son:
- Login " admin"
- Password "x"