Sistemas Electrónicos
Ingeniería y Diseño

Sistemas Electrónicos Ingeniería y Diseño
"Soluciones tecnológicas a la medida"
http://www.seid.com.mx email: seid@att.net.mx
Republica del Salvador 17, local 16. Col. Centro.
C.P. 06000. Deleg. Cuauhtémoc, México D.F.
Tel. (55) 5510 4081

Pasos para trasladar un programa en ensamblador hecho para un microcontrolador PIC de la gama media (instrucciones de 14 bits) a un PIC de la gama alta (instrucciones de 16 bits)

Nota 1: Este documento es válido para migrar programas con un número máximo de 128 variables
Nota 2: Los códigos que aparecen de ejemplo son para el PIC18F452 de la gama alta

1. Sustituya desde las directivas LIST e INCLUDE, el dispositivo (PIC) de la gama media, por el dispositivo de la gama alta que va a utilizar.

```
;*****PROCESADOR, SISTEMA NUMÉRICO PREDEFINIDO Y DEFINICIÓN DE LOS REGISTROS DE FUNCIÓN ESPECIAL Y SUS BITS***

        LIST        P=18F452        ;PIC DE LA GAMA ALTA CON 32 KBYTES DE MEMORIA DE PROGRAMA TIPO FLASH
        RADIX       HEX             ;SISTEMA HEXADECIMAL PREDEFINIDO
        INCLUDE     <P18F452.INC>   ;ARCHIVO FUENTE EN DONDE SE DEFINEN LAS DIRECCIONES QUE OCUPAN EN
                                    ;MEMORIA DE DATOS LOS REGISTROS DE FUNCIÓN ESPECIAL. TAMBIEN DEFINE LA
                                    ;DISPOSICIÓN DE LOS BITS DE ESTOS REGISTROS
```

2. Sustituya la configuración de los bits fusibles del PIC de la gama media, por la configuración de los bits fusibles de la gama alta, haciendo uso de la directiva CONFIG para modificar los registros CONFIG1H, CONFIG2L, CONFIG2H, CONFIG3H, CONFIG4L, CONFIG5L, CONFIG5H, CONFIG6L, CONFIG6H, CONFIG7L, y CONFIG7H de acuerdo a su aplicación.

```
;REGISTRO DE CONFIGURACION DE PROGRAMA 1H (_OSCS_OFF = LA FUENTE DE RELOJ DEL SISTEMA ESTA CONECTADA AL
;OSCILADOR PRINCIPAL Y NO ESTA CONECTADA A LA FUENTE DE RELOJ ALTERNA DE BAJA FRECUENCIA (TMR1), _HS_OSC =
;OSCILADOR EXERNO A CRISTAL DE 20MHz)

        __CONFIG    _CONFIG1H, _OSCS_OFF_1H & _HS_OSC_1H


;REGISTRO DE CONFIGURACION DE PROGRAMA 2L (_BOR_OFF = BROWN OUT RESET DESABILITADO Y POR LO TANTO NO
;AFECTA EL NIVEL DE CAIDA DE VOLTAJE A DETECTAR, _BORV_20 = 2.0V NIVEL DE CAIDA DE VOLTAJE A DETECTAR POR EL
;BROWN OUT RESET, _PWRT_ON = POWER ON RESET HABILITADO

        __CONFIG    _CONFIG2L, _BOR_OFF_2L & _BORV_20_2L & _PWRT_ON_2L


;REGISTRO DE CONFIGURACION DE PROGRAMA 2H (_WDT_ON = WATCH DOG TIMER HABILITADO, _WDTPS_1 = POSTESCALA
1:1 ;DEL WDT)

        __CONFIG    _CONFIG2H, _WDT_ON_2H & _WDTPS_1_2H


;REGISTRO DE CONFIGURACION DE PROGRAMA 3H (_CCP2MX_ON = LA ENTRADA O SALIDA DEL MODULO CCP2 ES POR RC1,
;PERO COMO EN ESTA APLICACIÓN NO SE UTILIZA EL MODULO DE CAPTURA (CCP2) NO INTERESA QUE LA FUNCION DE RC1
;ESTE MULTIPLEXADA CON LA DEL MODULO DE CAPTURA)

        __CONFIG    _CONFIG3H, _CCP2MX_ON_3H


;REGISTRO DE CONFIGURACION DE PROGRAMA 4L (_STVR_OFF = EL EQUIPO NO SE RESETEARA CUANDO EL STACK POINTER
;ESTE LLENO O SE HALLA DESBORDADO, _LVP_OFF = NO SE REQUIERE PROGRAMAR AL PIC CON UN BAJO VOLTAJE DE
;PROGRAMACION (5V), _DEBUG_OFF = NO SE REQUIERE CORRER EL DEBUGER POR COMPUTADORA (PC) ASI QUE RB6 Y RB7
;SE CONFIGURAN COMO LINEAS DE ENTRADA/SALIADA DE PROPOSITO GENERAL)

        __CONFIG    _CONFIG4L, _STVR_OFF_4L & _LVP_OFF_4L & _DEBUG_OFF_4L


;REGISTRO DE CONFIGURACION DE PROGRAMA 5L (_CP0_ON = _CP1_ON = _CP2_ON = _CP3_ON = TODOS LOS BLOQUES DE
;MEMORIA DE PROGRAMA ESTAN PROTEGIDOS)
```

```
    __CONFIG    _CONFIG5L, _CP0_ON_5L & _CP1_ON_5L & _CP2_ON_5L & _CP3_ON_5L

;REGISTRO DE CONFIGURACION DE PROGRAMA 5H (_CPB_ON = CODIGO DE BLOQUE DE ARRANQUE PROTEGIDO, _CPD_OFF =
;MEMORIA EEPROM DE DATOS DESPROTEGIDA)

    __CONFIG    _CONFIG5H, _CPB_ON_5H & _CPD_OFF_5H

;REGISTRO DE CONFIGURACION DE PROGRAMA 6L (_WRT0_ON = _WRT1_ON = _WRT2_ON = _WRT3_ON = TODOS LOS BLOQUES
;DE MEMORIA DE PROGRAMA ESTAN PROTEGIDOS CONTRA ESCRITURA (PROTEGIDOS CONTRA ESCRITURA POR TABLA))

    __CONFIG    _CONFIG6L, _WRT0_ON_6L & _WRT1_ON_6L & _WRT2_ON_6L & _WRT3_ON_6L

;REGISTRO DE CONFIGURACION DE PROGRAMA 6H (_WRTC_ON = REGISTROS DE CONFIGURACION PROTEGIDOS CONTRA
;ESCRITURA, WRTB_ON = BLOCK DE ARRANQUE PROTEGIDO CONTRA ESCRITURA, _WRTD_OFF = MEMORIA EEPROM DE
;DATOS DESPROTEGIDA DE ESCRITURA

    __CONFIG    _CONFIG6H, _WRTC_ON_6H & _WRTB_ON_6H & _WRTD_OFF_6H

;REGISTRO DE CONFIGURACION DE PROGRAMA 7L (_EBTR0_ON = _EBTR1_ON = _EBTR2_ON = _EBTR3_OFF = TODOS LOS
;BLOQUES DE MEMORIA DE PROGRAMA DESPROTEGIDOS CONTRA LECTURA DE TABLAS EJECUTADAS DESDE OTROS
;BLOQUES)

    __CONFIG    _CONFIG7L, _EBTR0_OFF_7L & _EBTR1_OFF_7L & _EBTR2_OFF_7L & _EBTR3_OFF_7L

;REGISTRO DE CONFIGURACION DE PROGRAMA 7H (_EBTRB_OFF = BLOQUE DE ARRANQUE DESPROTEGIDO CONTRA LECTURA
;DE TABLAS EJECUTADAS DESDE OTROS BLOQUES) OJO SI ESTE BIT SE ENCIENDE (_EBTRB_ON) AL MOMENTO DE
;PROGTAMAR EL PIC MANADARA UN MENSAJE QUE NO SE GRABO ADECUADAMENTE

    __CONFIG    _CONFIG7H, _EBTRB_OFF_7H
```

3. Declare los registros de propósito general (variables) a partir de la dirección 00h y no desde la 20h como se hacia anteriormente; esto lo puede realizar haciendo uso de la directiva CBLOCK. Este cambio afectará la o las rutinas que trasladan bloques de memoria entre la EEPROM y la SRAM y viceversa, ya que se deberán redefinir las direcciones de inicio y final de traslado de datos. Esto se detalla mas adelante en el manejo de memoria de datos por direccionamiento indirecto.

4. Incluya las variables que son accesibles desde cualquier banco y que se definen a partir de la dirección 70h en la gama media, al nuevo bloque de variables que se acaba de definir y que inicia en la dirección 00h, cuidando que la última variable no rebase la dirección 7Fh, lo cual será fundamental para garantizar el correcto traslado del programa de la gama media a la gama alta. Para esto último, una el segundo bloque de variables que inicia en la dirección 70h (variables accesibles desde cualquier banco en la gama media), al nuevo bloque que se acaba de definir y que inicia en la dirección 00h, eliminando la directiva ENDC del primer bloque y la directiva CBLOCK del segundo bloque. Esto hará que todas las variables comprendidas entre la dirección 00h y la dirección 7Fh del nuevo bloque de memoria, queden ubicadas en las primeras 128 direcciones del banco 0, y por lo tanto se podrá acceder a ellas desde cualquier banco, ***si y solo si el campo 'a' de la instrucción valga '0' (access bank),*** no importando en cual de los 16 bancos se este trabajando al momento de acceder a ellas.

   Cuando el número de variables no son más de 128, todas se podrán declarar en un solo bloque de memoria de la dirección 00h hasta la 7Fh dentro del banco 0, en el espacio destinado a los registros de propósito general, y se podrá acceder a ellas sin necesidad de hacer uso del registro BSR<3:0> para seleccionar uno de los 16 bancos de memoria disponibles, sólo dándole al campo 'a' de la instrucción el valor '0', lo cual se da por default en

la instrucción si es que no se declara este valor (tal y como viene el código para la gama media), facilitando con esto aún más el traslado del programa.

Si el número de variables necesarias para la ejecución de un programa es mayor a 128, garantice que las variables que deben ser accesibles desde cualquier banco queden comprendidas de la dirección 00h a la dirección 7Fh del banco 0, para que se pueda acceder a ellas por medio del ***access bank,*** lo cual evitará la declaración de dos bloques de memoria como se hace para la gama media. También tome en cuenta que en este caso tendrá que hacer uso del registro BSR<3:0> para acceder a las variables declaradas en otros bancos.

```
;*****DECLARACIÓN DE REGISTROS DE PROPÓSITO GENERAL*************************************************************************************

                    CBLOCK   00H
SP_ENT                              ;00H
SP_FLOT
H_ENT
H_FLOT
AL_SUP_ENT
AL_SUP_FLOT
AL_INF_ENT
AL_INF_FLOT
BANDERA_OFFS
OFFSET_LM35
CLAVE_4
CLAVE_3
CLAVE_2
CLAVE_1
RET_REL
T_MUESTREO
BANDERA_CANDADO                     ;10H
BANDERA_BUZZER
REMANENCIA
BANDERA_ON_OFF
BANDERA_MODO
BRILLO
COMUNICACION
DIRECCION_RED
MODELO                              18H

DECRE1                              ;19H, USADO EN LAS SUBRUTINAS DE TIEMPO
DECRE2                              ;USADO EN LAS SUBRUTINAS DE TIEMPO
DECRE3                              ;USADO EN LAS SUBRUTINAS DE TIEMPO
DECRE4                              ;USADO EN LAS SUBRUTINAS DE TIEMPO
DECRE5                              ;1DH, USADO EN LAS SUBRUTINAS DE TIEMPO
                    ENDC
```

5. Si declara en la EEPROM valores predeterminados (valores de fábrica), esto lo deberá hacer a partir de la dirección F00000h en lugar de la dirección 2100h como se hacía en la gama media, mediante el uso de la directiva ORG. Cuando lo haga declare 2 datos por cada directiva "DE" en lugar de uno sólo como se declaraba anteriormente, ya que cada directiva 'DE' en la gama alta representa un campo de 16 bits en el cual caben 2 datos de 8 bits; es decir que cada directiva DE representa el contenido de dos direcciones de memoria EEPOM. También puede declarar más de dos datos por cada directiva 'DE', pero se recomienda hacerlo en pares para evitar errores de correspondencia al momento de realizar algoritmos de traslados de bloques de memoria entre la EEPROM y la SRAM y viceversa.

```
;*****PARAMETROS DE CONFIGURACION INICIAL DEL EQUIPO GUARDADOS EN LA EEPROM*************************************************

                    ORG     0F00000H

                    DE      D'26',D'0'      ;SET POINT (PARTE ENTERA Y PARTE FLOTANTE)
                    DE      D'3',D'0'       ;HISTERESIS (PARTE ENTERA Y PARTE FLOTANTE)
                    DE      D'32',D'0'      ;ALARMA SUPERIOR (PARTE ENTERA Y PARTE FLOTANTE)
                    DE      D'21',D'0'      ;ALARMA INFERIOR (PARTE ENTERA Y PARTE FLOTANTE)
                    DE      D'0',D'0'       ;SIGNO DEL OFFSET Y OFFSET (+0)
```

Sistemas Electrónicos Ingeniería y Diseño
"Soluciones tecnológicas a la medida"
http://www.seid.com.mx email: seid@att.net.mx
Republica del Salvador 17, local 16. Col. Centro.
C.P. 06000. Deleg. Cuauhtémoc, México D.F.
Tel. (55) 5510 4081

```
DE      D'0',D'0',D'0',D'0'  ;CLAVE DE ACCESO AL MENU DE CONFIGURACIONES (4 DIGITOS)
DE      D'0',D'50'          ;RETARDO DEL RELEVADOR DE CONTROL
                            ;TIEMPO DE MUESTREO (50X10mS=500mS)
DE      D'0',D'0'           ;BANDERA CANDADO (0=SIN CANDADO)
                            ;BANDERA BUZZER (0=BUZZER DESHABILITADO, 1=BUZZER HABILITADO)
DE      D'1',D'0'           ;REMANENCIA (1=CON REMANENCIA)
                            ;BANDERA DE ENCENDIDO/APAGADO (0=EQUIPO APAGADO)
DE      D'1',D'1'           ;MODO DE TRABAJO (1=MODO CALEFACTOR)
                            ;BRILLO DE PANTALLA (1=50%)
DE      D'0',D'1'           ;COMUNICACION (0=SIN COMUNICACION, 1=CON COMUNICACION)
                            ;DIRECCION EN RED RS-485
DE      'E',D'0'            ;MODELO DEL EQUIPO E=TERMOCONTROL D=SOLO ALARMA A=SOLO CONTROL
                            ;BYTE (DATO) DE RELLENO
```

6. Elimine todos los cambios de banco en el programa precediendo con (;) las instrucciones BCF STATUS,RP0, BSF STATUS,RP0, BCF STATUS,RP1 o BSF STATUS,RP1; o si utilizo etiquetas de sustitución de código como BANCO_0 o BANCO_1, también deberá precederlas con (;) para que el ensamblador las detecte como comentarios y sean ignoradas a la hora de ensamblar el programa.

7. Elimine todos los cambios de pagina en el programa precediendo con (;) las instrucciones BCF PCLATH,3, BSF PCLATH,3, BCF PCLATH,4 o BSF PCLATH,4; o si utilizo etiquetas de sustitución de código como PAG0_A_PAG1, PAG0_A_PAG2, etc. también deberá precederlas con (;) para que el ensamblador las detecte como comentarios y sean ignoradas a la hora de ensamblar el código.

8. Declare la dirección 0008h como vector de interrupciones en lugar de la dirección 0004h como se hacía para la gama media. En la gama alta el vector de interrupciones de alta prioridad se encuentra ubicado en la dirección 0008h, y aunque esta gama también cuenta con un vector de interrupciones de baja prioridad (dirección 0018h), este vector no se definirá debido a que no se manejaran interrupciones de baja prioridad, porque se trabajara al microcontrolador en modo de compatibilidad con la gama media, y debido a esto todas las interrupciones se dirigirán al vector de interrupciones de alta prioridad. Una vez que se lleve a cabo el traslado correcto del programa, el usuario podrá asignar jerarquías a las interrupciones que maneje y solo en ese caso deberá declarar el vector de interrupciones de baja prioridad en el programa.

9. Modifique en el vector de interrupciones el procedimiento para respaldar el contenido de las variables que no se deben afectar al momento en que se da una interrupción, para que vuelvan al programa principal con los valores que tenían al momento en que se genero la interrupción. En la gama media los valores que se respaldaban al momento de generarse una interrupción eran los contenidos de los registros W, STATUS y PCLATH, tal y como se muestra abajo. Ahora bien debido a que en la gama alta no existe paginación de la memoria de programa, entonces no es necesario respaldar el contenido del PCLATH para saber en que página se encontraba el contador de programa (PC) al momento de darse la interrupción; por lo que lo único que se deberá respaldar en este caso, son los contenidos de los registros W, STATUS y BSR. El contenido de W se resguarda, por que es el registro principal de trabajo sobre el cual se efectúan la mayoría de las operaciones, y en consecuencia se alterará durante la ejecución del programa de atención de interrupciones. El contenido del STATUS también se respalda, por que en él se encuentran bits bandera que se ven afectados por la ejecución de operaciones aritméticas y lógicas, mismas que se pueden ejecutar en el programa de atención de interrupciones y por lo tanto afectar estos bits bandera. Finalmente

el contenido del registro BSR se debe respaldar al momento en que se da una interrupción, por que en él se encuentra definido, el banco de memoria de datos en donde se estaba trabajando previo a la interrupción. Los contenidos que los registros W, STATUS y BSR tenían previos a una interrupción, deberán recuperarse antes de salir del programa de atención de interrupciones, para que el programa principal se siga ejecutando de forma correcta.

```
        ORG.    0004H

        MOVWF   W_RESP              ;respalda a W
        SWAPF   STATUS,W            ;respalda a STATUS
        MOVWF   STATUS_RESP
        MOVF    PCLATH,W            ;respalda a PCLATH
        MOVWF   PCLATH_RESP
        BSF     PCLATH,4            ;define la pagina donde se encuentra la interrupción
        BSF     PCLATH,3
        BSF     STATUS,RP1          ;define el banco en que se va a trabajar la interrupción
        BSF     STATUS,RP0
        GOTO    INTERRUPCION        ;ve a la interrupción
```

Procedimiento para respaldar el contenido de los registros W, STATUS y PCLATH en el momento en que se genera una interrupción en la gama media

```
        ORG. 0008H

        MOVWF   W_RESP              ;respalda a W
        MOVFF   STATUS,STATUS_RESP  ;respalda a STATUS
        MOVFF   BSR,BSR_RESP        ;respalda a BSR
        GOTO    INTERRUPCION        ;ve a la interrupción
```

Procedimiento para respaldar el contenido de los registros W, STATUS y BSR en el momento en que se genera una interrupción en la gama alta

10. Modifique en el programa de atención de interrupciones, el procedimiento para recuperar el contenido de los registros respaldados en el vector de interrupciones (W, STATUS y PCLATH para la gama media y W, STATUS y BSR para la gama alta) antes de salir de la interrupción. Ver código abajo

```
SALIR   MOVF    PCLATH_RESP,W       ;recupera a PCLATH
        MOVWF   PCLATH
        SWAPF   STATUS_RESP,W       ;recupera a STATUS
        MOVWF   STATUS
        SWAPF   W_RESP,F            ;recupera a W
        SWAPF   W_RESP,W
        RETFIE                      ;regreso de interrupción
```

Procedimiento para recuperar el contenido de los registros W, STATUS y PCLATH antes de salir de la interrupción para la gama media

```
SALIR   MOVFF   BSR_RESP,BSR        ;recupera a BSR
        MOVF    W_RESP,W            ;recupera W
        MOVFF   STATUS_RESP,STATUS  ;recupera a STATUS
        RETFIE                      ;regreso de interrupción
```

Procedimiento para recuperar el contenido de los registros W, STATUS y BSR antes de salir de la interrupción para la gama alta

11. Si requiere habilitar las PULL-UPS del puerto B, hágalo sustituyendo el código: BCF OPTION_REG,NOT_RBPU por: BCF INTCON2,NOT_RBPU

```
        BCF     OPTION_REG,NOT_RBPU  ;ASI SE HACIA PARA LA GAMA MEDIA (PIC16F87XA)
```

```
        BCF     INTCON2,NOT_RBPU    ;ASI SE HACE PARA LA GAMA ALTA (PIC18FXX2)
```

12. Elimine el código donde se asigna el prescaler del TIMER 0 al WDT, precediendo con (;) el código correspondiente (BSF OPTION_REG,PSA). Esto se hace por que en la gama alta, el WDT no comparte el preescaler con el TMR0, ya que tiene su propio postescaler.

13. Elimine el código que selecciona el valor de preescala del TIMER 0 que se asignó al WDT, porque para la gama alta el valor de postescala para el WDT se asigna desde los bits de configuración (consulte punto 2).

```
;ASIGNACIÓN DEL PRESCALER AL WDT

;       BSF     OPTION_REG,PSA      ;ASI SE HACIA PARA LA GAMA MEDIA (PIC16F87XA)
                                    ;PARA LA GAMA ALTA (PIC18FXX2) EL WDT TIENE UN POSTESCALER
                                    ;INDEPENDIENTE Y POR LO TANTO NO ES NECESARIO ASIGNARLO

;SELECCIÓN DE LA PREESCALA (1:1) PARA OBTENER 18 MILISEGUNDOS PARA EL DESBORDAMINETO DEL WDT (BASE
DEL ;TIMER 18ms)

;       BCF     OPTION_REG,PS2      ;ASI SE SELECCIONABA UNA PRESCALA 1:1 PARA LA GAMA MEDIA
;       BCF     OPTION_REG,PS1      ;(PIC16F87XA). PARA LA GAMA ALTA (PIC18FXX2) LA SELECCION
;       BCF     OPTION_REG,PS0      ;DE LA POSTESCALA SE HACE DESDE LOS BITS DE CONFIGURACION
```

14. Elimine todas las directivas ORG que dan inicio a los cambios de página, precediendo con (;) la o las directivas ORG implicadas. Esto se hace por que en la gama alta no existe la paginación de memoria de programa y en consecuencia la definición del inicio de cada página ya no es necesario. La eliminación de la dirección de memoria de inicio de cada página también evitara en algunos casos, que a la hora de ensamblar el código del programa para los dispositivo de la gama alta, se generen errores de sobre escritura, ya que es probable que los códigos de programa se sobrescriban, por que en la gama alta cada instrucción consta de 2 o 4 bytes, y como la memoria de programa esta direccionada en bytes y no por palabras (Word), entonces el contador de programa (PC) se incrementa 2 o 4 veces por instrucción en lugar de 1 o 2 veces como sucedía en la gama media; lo que hace muy probable la posibilidad que se sobrescriban localidades de memoria en la siguiente página; esto claro esta si no se eliminan las directivas ORG correspondientes al inicio de cada página.

```
;       ORG     0800H               ;INICIA PAGINA 1
;       ORG     1000H               ;INICIA PAGINA 2
;       ORG     1800H               ;INICIA PAGINA 3
```

15. Si el programa a transferir cuenta con tablas. Estas se deberán colocar al final del programa (antes de la directiva END) y su correspondiente dirección de inicio definida por la directiva ORG deberá ponerse como comentario precediéndola con (;), para evitar errores de sobre escritura. En los puntos 19 y 21 se da por terminado el tema de lectura de tablas

```
;*****TABLA DE CARACTERES 1 (256 CARACTERES)*************************************************************************************

;       ORG     0EFCH

TABLA   MOVLW   0FH                 ;CARGA AL PCLATH CON LA PAGINA DONDE SE ENCUENTRA LA TABLA DE
        MOVWF   PCLATH              ;CACTERES
        MOVF    DESBA,W             ;AL SER EL PCL EL DESTINO DE ESTA INSTRUCCIÓN , LA PARTE ALTA DEL
        ADDWF   PCL,F               ;CONTADOR DE PROGRAMA SE CARGA CON ELCONTENIDO DEL PCLATH Y SU
                                    ;PARTE BAJA SE CARGA CON EL CONTENIDO DEL PCL (OFFSET)

        DT      "GPO ORGO"          ;CUADRO_1 (0D)

        DT      "VENTAS: "          ;CUADRO_2 (8D)
```

```
        DT      "53654669"              ;CUADRO_3 (16D)
        DT      "COMU:   "              ;CUADRO_4 (24D)

        DT      05,"APAGAR?"            ;CUADRO_5 (32D)

        DT      05,"AJ. PC?"            ;CUADRO_6 (40D)

        DT      "PC     ",1B            ;CUADRO_7 (48D)

        DT      "CA: ",0BF,"???"        ;CUADRO_8 (56D)

        DT      "BP:    %"              ;CUADRO_9 (64D)

        DT      "M0:     "              ;CUADRO_10 (72D)

        DT      "DIF    ",1B            ;CUADRO_11 (80D)

        DT      "AS     ",1B            ;CUADRO_12 (88D)

        DT      "AI     ",1B            ;CUADRO_13 (96D)

        DT      "RET:  s "              ;CUADRO_14 (104D)

        DT      "TM:    s"              ;CUADRO_15 (112D)

        DT      "OFFS    "              ;CUADRO_16 (120D)

        DT      "       "               ;CUADRO_17 (128D)

        DT      "REMA:   "              ;CUADRO_18 (136D)

        DT      05,"PONER CANDADO? "    ;CUADRO_19 (144D)
        DT      "NDADO? ",05,"PONER CA" ;CUADRO_19 (160D)

        DT      "CA: XXXX"              ;CUADRO_20 (176D)

        DT      05,"QUITAR CANDADO? "   ;CUADRO_21 (184D)
        DT      "NDADO? ",05,"QUITAR CA" ;CUADRO_21 (201D)

        DT      05,"CAMBIAR CLAVE? "    ;CUADRO_22 (218D)
        DT      "CLAVE? ",05,"CAMBIAR " ;CUADRO_22 (234D - 249D)

;       DT      "CA: 1234"              ;CUADRO_23 (258D)

        DT      "DIR.: "                ;CUADRO_24 (250D - 255D)


        END
```

16. Seleccione en el MPLAB el dispositivo de la gama alta que va a utilizar en su proyecto y ensamble el programa que acaba de adecuar. Si realizo correctamente los pasos que se describieron anteriormente, el ensamblador le arrojara errores y comentarios, que usted podrá solventar como comúnmente los hace; pero independientemente de esto a continuación se describen algunos posibles errores que el ensamblador puede arrojar, la causa que los provoca y su solución.

    ERROR: Símbolo no definido previamente: BSR_RESP
    CAUSA: No declaración de la variable BSR_RESP
    SOLUCION: Declare en el bloque de registros de propósito general definido por las directivas CBLOCK y ENDC la variable correspondiente

    ERROR: Símbolo no definido previamente: FSR
    CAUSA: En la gama alta existen más de un registro de función especial FSR y además cada uno de ellos cuenta con una parte alta y una parte baja (ej. FSR0H, FSR0L, FSR1H, FSR1L, FSR2H, FSR2L)

ERROR: Símbolo no definido previamente: INDF
CAUSA: En la gama alta existen más de un registro de función especial INDF (ej. INDF0, INDF1, INDF2)
SOLUCION: Elija un juego de registros FSR e INDF (ej. FSR0H, FSR0L e INDF0), y ajuste su algoritmo para el manejo de memoria de datos por direccionamiento indirecto (Ver detalles en el punto 17)

ERROR: Símbolo no definido previamente: RRF FILE,F ó RRF FILE,W
ERROR: Símbolo no definido previamente: RLF FILE,F ó RLF FILE,W
CAUSA: Para la gama media, se tiene la opción de rotación con o sin el acarreo, así que para transferir correctamente el programa se debe elegir la instrucción adecuada.
SOLUCION: Sustituya la instrucción RRF por RRCF y la instrucción RLF por RLCF

17. Si tiene algoritmos para el manejo de memoria de datos por direccionamiento indirecto, donde se ven involucrados los registros FSR e INDF, bastara con limpiar la parte alta del registro FSR0 (CLRF FSR0H) al inicio del algoritmo y realizar las operaciones que normalmente se realizaban sobre el registro FSR ahora sobre FSR0L. Por otra parte las operaciones que se realizaban sobre el registro INDF ahora se deberán realizar sobre el registro INDF0. Si el algoritmo maneja traslado de datos entre las memorias de datos SRAM y EEPROM y viceversa, se deberá cuidar que las direcciones involucradas de la SRAM sean las mismas que se manejaban antes de realizar el traslado del programa, por que al redefinir la dirección de inicio de los registros de propósito general o variables (ver punto 3), es casi seguro que existirán errores de correspondencia a la hora de transferir bloques de memoria de datos. Por último, cuando se utilice direccionamiento indirecto para el manejo de datos provenientes de la memoria EEPROM, se debe definir el acceso a esta memoria colocando en '0' el bit CFGS del registro EECON1 antes de leer o escribir en ella, para de esta forma evitar el acceso a los bits de configuración (bit CFGS = '1') por error. Dos ejemplos particulares de este tipo de algoritmos son las subrutinas EEPROM_A_RAM y RAM_A_EEPROM.

```
EEPROM_A_RAM    BSF     STATUS,RP1      ;TRABAJA EN BANCO 2
                BCF     STATUS,RP0
                MOVLW   0FFH            ;CARGA A EEADR CON UNA DIRECCION ANTES DE LA PRIMERA A LEER
                MOVWF   EEADR           ;DE LA EEPROM
                MOVLW   20H             ;CARGA A FSR CON LA DIRECCION DE LA PRIMER VARIABLE DONDE
                MOVWF   FSR             ;SE DEPOSITARA EL CONTENIDO DE LA PRIMER DIRECCION LEIDA
                                        ; DESDE LA EEPROM
NEXT            CLRWDT
                INCF    EEADR,F         ;DIRECCION 00H EN EEPROM (VALOR INICIAL)
                BSF     STATUS,RP0      ;TRABAJA EN BANCO 3
                BCF     EECON1,EEPGD    ;ACCESA A LA MEMORIA EEPROM DE DATOS
                BSF     EECON1,RD       ;INICIA LA LECTURA DE LA EEPROM
                BCF     STATUS,RP0      ;TRABAJA EN BANCO 2
                MOVF    EEDATA,W        ;GUARDA EN W EL DATO RECUPERADO DE LA EEPROM Y
                MOVWF   INDF            ;DEPOSITALO EN INDF
                INCF    FSR,F
                MOVLW   39H             ;¿YA SE RECUPERARON TODOS LOS PARAMETROS DE CONFIGURACION
                XORWF   FSR,W           ;DE LA EEPROM?
                BTFSS   STATUS,Z
                GOTO    NEXT            ;NO
                BCF     STATUS,RP1      ;SI, TRABAJA EN BANCO 0
                RETURN
```

Recuperación de los parámetros de configuración del equipo mediante la lectura de la EEPROM para la gama media

```
EEPROM_A_RAM    CLRF    EEADR           ;CARGA A EEADR CON LA 1ER. DIRECCION EN EEPROM QUE SE VA A
                LEER
```

Sistemas Electrónicos Ingeniería y Diseño
"Soluciones tecnológicas a la medida"
http://www.seid.com.mx email: seid@att.net.mx
Republica del Salvador 17, local 16. Col. Centro.
C.P. 06000. Deleg. Cuauhtémoc, México D.F.
Tel. (55) 5510 4081

```
                CLRF    FSR0H       ;CARGA A FSR0 CON LA DIRECCION EN SRAM DE LA PRIMER VARIABLE
                CLRF    FSR0L       ;DONDE SE DEPOSITARA EL CONTENIDO DE LA PRIMER DIRECCION
                                    ;LEIDA DE LA EEPROM
NEXT            CLRWDT
                BCF     EECON1,EEPGD ;ACCESA A LA MEMORIA DE DATOS
                BCF     EECON1,CFGS ;Y NO A LOS BITS DE CONFIGURACION
                BSF     EECON1,RD   ;INICIA LA LECTURA DE LA EEPROM
                MOVFF   EEDATA,INDF0 ;DEPOSITA DE FORMA INDIRECTA EN LA DIRECCION EN SRAM A LA QUE
                                    ;APUNTA EL FSR0 EL DATO LEIDO DE LA EEPROM
                INCF    EEADR,F
                INCF    FSR0L,F
                MOVLW   19H         ;¿YA SE RECUPERARON TODOS LOS PARAMETROS DE CONFIGURACION
                CPFSEQ  FSR0L       ;DE LA EEPROM?
                GOTO    NEXT        ;NO
                RETURN              ;SI
```

Recuperación de los parámetros de configuración del equipo mediante la lectura de la EEPROM para la gama media

18. Siempre que realice algoritmos para grabar datos en la EEPROM, garantice el acceso a esta memoria colocando en '0' el bit CFGS del registro EECON1. Para nuestro ejemplo anexaremos la instrucción BCF EECON1,CFGS en la subrutina ESCRITEEPROM.

Nota: Al parecer si se invoca desde el programa de atención de interrupciones, una rutina como es la escritura a la EEPROM, en donde se deshabilitan momentáneamente las interrupciones generales no enmascaradas (BCF INTCON,GIE), para que no se vea interrumpido el proceso de escritura por la entrada de otra interrupción, causa problemas de pérdida del programa al momento de volver a habilitar las interrupciones generales (BSF INTCON,GIE), una vez terminado el proceso de escritura en la serie 18FXX2. Esto al parecer se debe a que otras fuentes de interrupción periféricas habilitadas por sus bits particulares, ocurren durante el proceso de escritura, y por lo tanto sus bits bandera de interrupción quedan levantados; y causan un conflicto al momento de volver a habilitar el bit GIE. Es por esta razón que se recomienda no volver a habilitar la interrupciones generales no enmascaradas, una vez concluido el proceso de grabación en la EEPROM como se hacía en la gama media, y simplemente dejar que se vuelva a habilitar el bit GIE de forma automática, al momento en que se sale del programa de atención de interrupciones mediante la ejecución de la instrucción RETFIE. En el código de ejemplo sólo se precedió con (;) la instrucción BSF INTCON,GIE para no habilitar las interrupciones generales no enmascaradas después de terminar el proceso de escritura a la EEPROM.

```
ESCRITEEPROM    MOVFF   EEDATA,EEDATA_RESP ;RESPALDA EL DATO QUE SE QUIERE ESCRIBIR EN LA EEPROM
RECTIFICA       MOVFF   EEDATA_RESP,EEDATA ;RECUPERA EL DATO QUE SE QUIERE ESCRIBIR EN LA EEPROM
                BCF     EECON1,EEPGD ;ACCESA A LA EEPROM DE DATOS
                BCF     EECON1,CFGS ;Y NO A LOS BITS DE CONFIGURACION
                BSF     EECON1,WREN ;HABILITA LA ESCRITURA
                BCF     INTCON,GIE  ;DESHABILITA LAS INTERRUPCIONES NO ENMASCARADAS
                MOVLW   55H         ;SECUENCIA DE INICIO DE ESCRITURA
                MOVWF   EECON2
                MOVLW   0AAH
                MOVWF   EECON2
                BSF     EECON1,WR   ;INICIA EL CICLO DE ESCRITURA
                BCF     EECON1,WREN ;DESHABILITA LA ESCRITURA
FINESCRIT       CLRWDT
                BTFSS   PIR2,EEIF   ;TERMINO EL CICLO DE ESCRITURA
                GOTO    FINESCRIT   ;NO, PREGUNTA DE NUEVO
                BCF     PIR2,EEIF   ;SI, LIMPIA LA BANDERA DE ESCRITURA COMPLETA (EEIF)
;               BSF     INTCON,GIE  ;Y HABILITA LAS INTERRUPCIONES NO ENMASCARADAS
VERIFICA        MOVF    EEDATA,W    ;INICIA VERIFICACIÓN DE ESCRITURA EN EEPROM
                BSF     EECON1,RD   ;LECTURA DE LA EEPROM
                CPFSEQ  EEDATA      ;¿SE ESCRIBIO CORRECTAMENTE EL DATO?
                GOTO    RECTIFICA   ;NO, VUELVE A ESCRIBIR EL DATO
                RETURN              ;SI
```

Subrutina para escribir y verificar la escritura a la EEPROM para la gama alta

19. Debido a que la lectura de tablas en la gama media se realiza por la suma de un offset (ADDWF PCL) al contador de programa (PC); para trasladar el programa íntegramente a la gama alta respetando este método, comenzaremos por describir brevemente este proceso.

Una lectura de tablas puede realizarse con una instrucción ADDWF PCL, un grupo de instrucciones RETLW 0xnn y un offset en la TABLA cargado previamente en el registro WREG antes de ejecutar su llamada. Para esto, la primera instrucción de la rutina es ADDWF PCL y la segunda un RETLW 0xnn, la cual regresará el valor 0xnn cuando se llame la función (rutina). De esta forma el valor del offset (valor de WREG), especificará el número de direcciones que el contador de programa (PC) deberá avanzar. Ahora bien, como en la gama alta, la memoria de programa esta direccionada en bytes y no en palabras (words) como en la gama media, entonces las instrucciones se almacenan en 2 o en 4 bytes, y por lo tanto el contador de programa avanza 2 o 4 bytes de acuerdo a la instrucción ejecutada. Es por esta razón que al utilizar este método para lectura de tablas en la gama alta, sólo se puede leer un byte por cada ejecución de la instrucción RETLW, ya que el contador de programa avanza 2 bytes por cada instrucción de este tipo; lo que trae como consecuencia un desperdicio de memoria de programa para la lectura de tablas por este método en esta gama.

Volviendo al ejemplo del punto 15, para terminar el traslado de un programa hecho para un dispositivo de la gama media que contenga lectura de tablas, es recomendable como ya se mencionó antes, colocar la tabla al final del programa y además destinarle un espacio definido dentro de esta memoria; así que suponiendo que el traslado de programa se va a hacer a un dispositivo de la gama alta con 32KBytes de memoria de programa como lo es el PIC18F452, se sugiere destinar la última cuarta parte de la memoria (dirección 6000h a dirección 7FFh) para la colocación de la tabla o tablas necesarias. Así que la dirección de inicio de la tabla 0EFCh que se muestra en el código del punto 15 y que está indicada por la directiva 0RG, ahora quedará definida por la misma directiva en la dirección 5FF8h y claro está que habrá de quitarle el (;) para que deje de ser un comentario. Ahora bien, como en la gama media se obtenían 256 datos de las 256 direcciones de la tabla original; y en la gama alta solo se puede obtener un dato por cada dos direcciones, entonces se tendrán que crear 3 tablas (ver código abajo) para poder obtener los 256 datos por lectura de tabla por este método, ya que el PCL rebasará el bloque de 256 direcciones y en consecuencia se tendrá que incrementar también el PCH, esto indirectamente a través del PCLATH. Algo más que se debe tomar en cuenta para la lectura de tablas por este método para la gama alta, es que se tiene que incrementar dos veces el offset en la tabla en lugar de una sola vez, como se hacía en la gama media, cada vez que se invoque la tabla para obtener el siguiente dato en la misma.

```
;*****TABLA1 DE CARACTERES (120 CARACTERES)**********************************************************************************************************

        ORG     5FF8H

TABLA1  MOVLW   60H             ;CARGA AL PCLATH CON LA PAGINA DONDE SE ENCUENTRA LA TABLA1 DE
        MOVWF   PCLATH          ;CARACTERES
        MOVF    DESBA,W         ;AL SER EL PCL EL DESTINO DE ESTA INSTRUCCIÓN , LA PARTE ALTA DEL
        ADDWF   PCL,F           ;CONTADOR DE PROGRAMA SE CARGA CON ELCONTENIDO DEL PCLATH Y SU
                                ;PARTE BAJA SE CARGA CON EL CONTENIDO DEL PCL (OFFSET)

        DT      "GPO ORGO"      ;CUADRO_1 (0D) DIR 6000H

        DT      "VENTAS: "      ;CUADRO_2 (16D)

        DT      "53654669"      ;CUADRO_3 (32D)

        DT      "COMU:   "      ;CUADRO_4 (48D)

        DT      05,"APAGAR?"    ;CUADRO_5 (64D)
```

```
            DT      05,"AJ. PC?"            ;CUADRO_6 (80D)

            DT      "PC     ",1B            ;CUADRO_7 (96D)

            DT      "CA: ",0BF,"???"        ;CUADRO_8 (112D)

            DT      "BP:    %"              ;CUADRO_9 (128D)

            DT      "M0:     "              ;CUADRO_10 (144D)

            DT      "DIF    ",1B            ;CUADRO_11 (160D)

            DT      "AS     ",1B            ;CUADRO_12 (176D)

            DT      "AI     ",1B            ;CUADRO_13 (192D)

            DT      "RET:  s "              ;CUADRO_14 (208D)

            DT      "TM:    s"              ;CUADRO_15 (224D - 239D)


;*****TABLA2 DE CARACTERES (114 CARACTERES)*****************************************************************************************

            ORG     60F8H

TABLA2      MOVLW   61H                     ;CARGA AL PCLATH CON LA PAGINA DONDE SE ENCUENTRA LA TABLA2 DE
            MOVWF   PCLATH                  ;CARACTERES
            MOVF    DESBA,W                 ;AL SER EL PCL EL DESTINO DE ESTA INSTRUCCIÓN , LA PARTE ALTA DEL
            ADDWF   PCL,F                   ;CONTADOR DE PROGRAMA SE CARGA CON ELCONTENIDO DEL PCLATH Y SU
                                            ;PARTE BAJA SE CARGA CON EL CONTENIDO DEL PCL (OFFSET)

            DT      "OFFS    "              ;CUADRO_16 (0D)

            DT      "       "               ;CUADRO_17 (16D)

            DT      "REMA:   "              ;CUADRO_18 (32D)

            DT      05,"PONER CANDADO? "    ;CUADRO_19 (48D)
            DT      "NDADO? ",05,"PONER CA" ;CUADRO_19 (80D)

            DT      "CA: XXXX"              ;CUADRO_20 (112D)

            DT      05,"QUITAR CANDADO? "   ;CUADRO_21 (128D)
            DT      "NDADO? ",05,"QUITAR CA" ;CUADRO_21 (162D)

            DT      05,"CAMBIAR CLAVE? "    ;CUADRO_22 (196D - 227D)


;*****TABLA3 DE CARACTERES (22 CARACTERES)*******************************************************************************************

            ORG     61F8H

TABLA3      MOVLW   62H                     ;CARGA AL PCLATH CON LA PAGINA DONDE SE ENCUENTRA LA TABLA3 DE
            MOVWF   PCLATH                  ;CARACTERES
            MOVF    DESBA,W                 ;AL SER EL PCL EL DESTINO DE ESTA INSTRUCCIÓN , LA PARTE ALTA DEL
            ADDWF   PCL,F                   ;CONTADOR DE PROGRAMA SE CARGA CON ELCONTENIDO DEL PCLATH Y SU
                                            ;PARTE BAJA SE CARGA CON EL CONTENIDO DEL PCL (OFFSET)

            DT      "CLAVE? ",05,"CAMBIAR " ;CUADRO_22 (0D - 31D)

;           DT      "CA: 1234"              ;CUADRO_23

            DT      "DIR.: "                ;CUADRO_24 (32D - 43D)


            END
```

20. Una vez trasladado el código a la nueva gama, puede eficientar el código haciendo uso de nuevas instrucciones como las siguientes:

CPFSEQ Esta instrucción compara el contenido de un registro (f) con el contenido del registro W y salta la siguiente instrucción si los contenidos son iguales. Con esta instrucción se pueden hacer comparaciones, sin hacer uso de la instrucción XORWF y el testeo del bit Z del registro STATUS como se hace en la gama media. Ej.

```
COMPARA   MOVLW          'D'          ;¿EL EQUIPO ES MODELO D?
          XORWF          MODELO,W
          BTFSS          STATUS,Z
          GOTO           MODELO_E     ;NO
          CONTINUA CODIGO             ;SI
```

Código para realizar comparaciones en la gama media

```
COMPARA   MOVLW          'D'          ;¿EL EQUIPO ES MODELO D?
          CPFSEQ         MODELO
          GOTO           MODELO_E     ;NO
          CONTINUA CODIGO             ;SI
```

Código para realizar comparaciones en la gama alta

MOVFF Esta instrucción mueve el contenido de un registro fuente a un registro destino. Con esta instrucción ya no es necesario el uso del registro W como intermediario para llevar a cabo esta acción como se hace en la gama media .Ej.

```
RESPALDA  MOVF           RCREG,W      ;RESPALDA EN COMANDO EL DATO QUE SE ACABA DE
          MOVWF          COMANDO      ;RECIBIR POR LA AUSART

          CONTINUA CODIGO
```

Código para mover el contenido de un registro fuente a un registro destino en la gama media

```
RESPALDA  MOVFF          RCREG,COMANDO ;RESPALDA EN COMANDO EL DATO QUE SE ACABA DE
                                       ;RECIBIR POR LA USART
          CONTINUA CODIGO
```

Código para mover el contenido de un registro fuente a un registro destino en la gama alta

TSTFSZ Esta instrucción compara el contenido de un registro (f) con cero y salta la siguiente instrucción si su contenido es cero. Con esta instrucción se pueden hacer testeos para saber si el contenido de un registro es cero, sin hacer uso de la instrucción MOVF y el testeo del bit Z del registro STATUS como se hace en la gama media. Ej.

```
REV_FLAG  MOVF           BANDERA,F    ;¿BANDERA = 0?
          BTFSS          STATUS,Z
          GOTO           NO_ES_CERO   ;NO
          CONTINUA CODIGO             ;SI
```

Código para determinar si el contenido de un registro es cero en la gama media

```
REV_FLAG  TSTFSZ         BANDERA      ;¿BANDERA = 0?
          GOTO           NO_ES_CERO   ;NO
          CONTINUA CODIGO             ;SI
```

Código para determinar si el contenido de un registro es cero en la gama alta

21. Como una mejora posterior al programa ya trasladado a la gama alta, llevaremos a cabo la adecuación del programa para que la lectura de tablas descrita en el punto 19, ahora se lleve a cabo por una operación TBLRD (LECTURA DE TABLAS), la cual permite al procesador de la gama alta mover bytes entre los espacios de memoria de programa y los espacios de memoria RAM de datos. Esta operación es más eficiente para la lectura de tablas que el método descrito en el punto 19, ya que permite almacenar dos datos en cada palabra de programa (localidad de instrucción) en lugar de sólo uno. Lo anterior se debe a que en la gama alta el espacio de memoria de programa es de 16 bits de ancho, mientras que el espacio de memoria de datos RAM es de 8 bits, y las operaciones para mover datos entre estos dos espacios de memoria se realizan a través del registro TABLAT de 8 bits. La operación TBLRD también requiere para su ejecución del TBLPTR (apuntador de tabla), el cual especifica la dirección del byte a leer en la memoria de programa y que se conforma por tres registros de función especial (SFRs), el registro superior del apuntador de tabla (TBLPTRU), el registro alto del apuntador de tabla (TBLPTRH) y el registro bajo del apuntador de tabla (TBLPTRL) y que juntos forman un apuntador de 22 bits de ancho; siendo los primeros 21 bits los que permiten al dispositivo direccionar hasta 2 MBytes en el espacio de memoria de programa y el bit 22 permite el acceso al ID del dispositivo, a él ID del usuario y a los bits de configuración. Una vez ejecutada la operación TBLRD, el dato leído desde la dirección del byte al que apunta el registro TBLPTR, es depositado en el registro TABLAT, desde donde podrá ser manipulado de acuerdo al algoritmo requerido. La operación de lectura se podrá repetir un byte a la vez.

El apuntador de tabla TBLPTR es utilizado tanto para la instrucción de lectura TBLRD, como para la instrucción de escritura TBLWR, las cuales pueden actualizarlo en una de las cuatro formas basadas en operaciones para el manejo de tablas. Estas operaciones se muestran en la tabla de abajo. Cabe mencionar que estas operaciones, sólo afectan los primeros 21 bits más bajos del TBLPTR. Cuando se ejecuta una instrucción TBLRD, los 22 bits del apuntador de tabla, determinan cual byte es leído de la memoria de programa en el registro TABLAT.

Ejemplo	Operación en el apuntador de tabla
TBLRD*	TBLPTR no se modifica
TBLWT*	
TBLRD*+	TBLPTR se incrementa despues de la lectura o la escritura
TBLWT*+	
TBLRD*-	TBLPTR se decrementa despues de la lectura o la escritura
TBLWT*-	
TBLRD+*	TBLPTR se incrementa antes de la lectura o la escritura
TBLWT+*	

Tabla 1: Actualización del registro TBLPTR en base a la operación para el manejo de tablas

Volviendo al ejemplo del punto 19, la tabla quedará definida de la siguiente forma para poder ser leída por una operación TBLRD. Entre los cambios más significativos se puede observar, que las tres tablas anteriores se integraron en una sola, además que la directiva DT se sustituyó por la directiva DE para poder reservar palabras de memoria de 8 bits, en lugar de la generación de instrucciones RETLW como lo hace la directiva DT.

```
;*****TABLA DE CARACTERES (256 CARACTERES)*********************************************************************************************
```

```
        ORG     6000H
TABLA   DE      "GPO ORGO"              ;6000H CUADRO_1
        DE      "VENTAS: "              ;6008H CUADRO_2
        DE      "53654669"              ;6010H CUADRO_3
        DE      "COMU:   "              ;6018H CUADRO_4
        DE      05,"APAGAR?"            ;6020H CUADRO_5
        DE      05,"AJ. PC?"            ;6028H CUADRO_6
        DE      "PC     ",1B            ;6030H CUADRO_7
        DE      "CA: ",0BF,"???"        ;6038H CUADRO_8
        DE      "BP:    %"              ;6040H CUADRO_9
        DE      "M0:    "               ;6048H CUADRO_10
        DE      "DIF    ",1B            ;6050H CUADRO_11
        DE      "AS     ",1B            ;6058H CUADRO_12
        DE      "AI     ",1B            ;6060H CUADRO_13
        DE      "RET:  s "              ;6068H CUADRO_14
        DE      "TM:    s"              ;6070H CUADRO_15
        DE      "OFFS    "              ;6078H CUADRO_16
        DE      "       "               ;6080H CUADRO_17
        DE      "REMA:   "              ;6088H CUADRO_18
        DE      05,"PONER CANDADO? "    ;6090H CUADRO_19 (48D)
        DE      "NDADO? ",05,"PONER CA" ;60A0H CUADRO_19 (80D)
        DE      "CA: XXXX"              ;60B0H CUADRO_20
        DE      05,"QUITAR CANDADO? "   ;60B8H CUADRO_21 (128D)
        DE      "NDADO? ",05,"QUITAR CA" ;60C9H CUADRO_21 (162D)
        DE      05,"CAMBIAR CLAVE? "    ;60DAH CUADRO_22 (196D - 227D)
        DE      "CLAVE? ",05,"CAMBIAR " ;60EAH CUADRO_22 (0D - 31D)
;       DE      "CA: 1234"              ;CUADRO_23
        DE      "DIR.: "                ;60FAH CUADRO_24

        END
```

Abajo se presenta como ejemplo, los códigos de las rutinas para leer los datos que conforman el CUADRO_1 y desplegarlos en dos pantallas alfanuméricas (PD 4437) de 4 caracteres cada una, conectadas en cascada.

```
;*****SUBRUTINA PARA DESPLEGAR EL CUADRO_1 EN LA PANTALLA*************************************************************************

CUADRO_1    MOVLW   60H         ;POSICIONA EL APUNTADOR DE TABLA DONDE INICIA EL MENSAJE
            MOVWF   TBLPTRH     ;QUE SE QUIERE MOSTRAR EN EL DISPLAY
            MOVLW   00H
            MOVWF   TBLPTRL
            CALL    CADENA1A    ;RUTINA PARA EL DESPLIEGUE DE LA CADENA DE CARACTERES EN LA PANTALLA
            RETURN
```

Sistemas Electrónicos Ingeniería y Diseño
"Soluciones tecnológicas a la medida"
http://www.seid.com.mx email: seid@att.net.mx
Republica del Salvador 17, local 16. Col. Centro.
C.P. 06000. Deleg. Cuauhtémoc, México D.F.
Tel. (55) 5510 4081

```
;*****SUBRUTINA PARA ENVIAR A LAS PANTALLAS UNA CADENA DE 8 CARACTERES LEIDOS DESDE LA TABLA**************************

CADENA1A    MOVLW   D'4'         ;CARGA A CONT_DIR CON UNA DIRECCIÓN MAYOR EN UNO A LA
            MOVWF   CONT_DIR     ;DIRECCION DEL DIGITO 3 DEL DISPLAY PD4437
            MOVLW   D'4'         ;CADENA DE 4 CARACTERES PARA LA PANTALLA 1
            MOVWF   XCAR
MENSAJE1    TBLRD*+              ;INCREMENTA TBLRD DESPUES DE LA LECTURA DE TABLA
            MOVF    TABLAT,W     ;DESPLIEGA EL CARACTER LEIDO EN LA PANTALLA 1
            CALL    CARACTER1A
            DECFSZ  XCAR,F       ;¿YA SE DESPLEGARON LOS 4 CARACTERES EN LA PANTALLA 1?
            GOTO    MENSAJE1     ;NO

CADENA1A2   MOVLW   D'4'         ;SI, CADENA DE 4 CARACTERES PARA LA PANTALLA 2
            MOVWF   XCAR
MENSAJE2    TBLRD*+              ;INCREMENTA TBLRD DESPUES DE LA LECTURA DE TABLA
            MOVF    TABLAT,W     ;DESPLIEGA EL CARACTER LEIDO EN LA PANTALLA 2
            CALL    CARACTER2A
            DECFSZ  XCAR,F       ;¿YA SE DESPLEGARON LOS 4 CARACTERES EN LA PANTALLA 2?
            GOTO    MENSAJE2     ;NO
            RETURN               ;SI

;*****SUBRUTINA PARA ESCRIBIR UN CARACTER EN LA PANTALLA 1*****************************************************************

CARACTER1A  MOVWF   DATOS        ;COLOCA EN EL BUS DATOS EL CODIGO DEL CARACTER
            DECF    CONT_DIR,W   ;COLOCA EN EL BUS DIRECCIONES LA DIRECCION DONDE SE VA A ESCRIBIR
            MOVWF               CONT_DIR
            MOVWF               DIRECCION
            BSF                 A2_4437
            BSF                 CE1D1_4437
            BCF                 WR_4437
            BSF                 WR_4437
            BCF                 CE1D1_4437
            RETURN

;*****SUBRUTINA PARA ESCRIBIR UN CARACTER EN LA PANTALLA 2*****************************************************************

CARACTER2A  MOVWF   DATOS        ;COLOCA EN EL BUS DATOS EL CODIGO DEL CARACTER
            DECF    CONT_DIR,W   ;COLOCA EN EL BUS DIRECCIONES LA DIRECCION DONDE SE VA A ESCRIBIR
            MOVWF   CONT_DIR
            MOVWF   DIRECCION
            BSF     A2_4437
            BSF     CE1D2_4437
            BCF     WR_4437
            BSF     WR_4437
            BCF     CE1D2_4437
            RETURN
```