UNIVERSIDAD DEL VALLE DE GUATEMALA

Programación de microcontroladores - SECCIÓN - 22 - 2025

PROYECTO – Reloj

Daniela Alexandra Moreira Cruz, 23841

GUATEMALA, 21 de marzo de 2025

Índice

Configuración, registros y variables

```
.include "M328PDEF.inc"
//definición de variables útiles
.equ    T0VALUE = 6 ;numero en el que debe empezar a contar T0
.equ    T1VALUE = 0x0BCD ;numero en el que debe empezar a contar T0
.equ    MAX_UNI = 9 ;numero para el overflow unidades
.equ    MAX_DEC = 5 ;numero para el overflow decenas
.equ    MAXT0   = 5
.equ    MODOS = 6 ; número máximo de modos
.equ    CICLO = 1 ; número de ciclos del timer 1 que deben cumplirse
.def    CONTADORT0= R17; contador para llevar el registro de los transistores de los display
.def    MODO    = R18 ; variable para el contdor de los modos
.def    CONTADORT1L= R19
.def    S_DISPLAY = R20
.def    CONTADOR_TIEMPO = R21
.def    DISPLAYS = R22
.def    CONTADOR_BOTONES= R23
.def    ACCION = R24
.def    ALARMA_V = R25
.dseg
.org    SRAM_START
MINUTO:             .byte   1 ; para registar que ya paso un min y debe cambiar umin
UMIN:               .byte   1 ; la variable que guarda el conteo de unidades de minutos
DMIN:               .byte   1 ; la variable que guarda el conteo de decenas de minutos
```

Esta es la definición de todos los registros utilizados en el programa y algunos valores regurentes utilizados en el programa, como límites o los valores de configuración de los timers. También se puede ver que se utilizó es espacio de la SRAM para almacenar datos de variables utilizadas en las funciones.

```
.cseg
.org 0x0000
    RJMP SETUP
.org PCI1addr // para el pin change
    JMP ISR_PCINT1
.org    0x001A
    RJMP    TMR1_ISR ; para el timer 1
.org OVF0addr // Para el timer 0
    JMP TMR0_ISR ; interrupción del Timer 0
```

Estos son los vectores de interrupción de pin change, interrupciones de overflow del timer 0 y 1. Cada vez que se genere alguna de las acciones que activen las interrupciones el programa saltará a la línea indicada en el vector de interrupción.

```
/************* INTERRUPCIONES  ********/
//timer 0
LDI     R16, (1 << TOIE0)
STS     TIMSK0, R16
//timer 1
LDI     R16, (1<<TOIE1)
STS     TIMSK1, R16


// Habilitar las interrupciones para el antirebote
LDI R16, (1<<PCINT8) | (1<<PCINT9) | (1<<PCINT10) | (1<<PCINT12) // Habilitar pin 0, pin 1 y pin 2
STS PCMSK1, R16       // Cargar a PCMSK1
LDI R16, (1 << PCIE1) // Habilitar interrupciones para el pin C
STS PCICR, R16
```

Aquí se habilitan las interrupciones para pin change en los pines 0 ,1 y 3. También se habilitan las interrupciones de overflow de los timer utilizados.

```
// Configuración MCU
SETUP:
    CLI
    /**************COFIGURACION DE PRESCALER********/
    // Configuración de prescaler inicial
    LDI     R16, (1 << CLKPCE)
    STS     CLKPR, R16 // Habilitar cambio de PRESCALER
    LDI     R16, 0b00000100
    STS     CLKPR, R16 // Configurar Prescaler en 1MHz

    /*************COFIGURACION DE PINES ********/
    // Configurar PB como salidas
    LDI     R16, 0xFF
    OUT     DDRB, R16       // Puerto B como salida
    LDI     R16, 0x00
    OUT     PORTB, R16      // El puerto B conduce cero lógico.

    //Configurar el puerto D como salidas, estabecerlo en apagado
    LDI     R16, 0xFF
    OUT     DDRD, R16 // Setear puerto D como salida
    LDI     R16, 0x00
    OUT     PORTD, R16 //Todos los bits en apagado

// Configurar PC3 y PC5 como salidas, PC0-PC2 como entradas
LDI R16, 0b00101000       ; 00011000 - PC3 y PC5 como salidas, el resto entradas
OUT DDRC, R16      ; Escribir en el registro DDRC

// Activar pull-ups en PC0, PC1, PC2 y PC3
LDI R16, 0b00010111      ; 00000111 - Habilita pull-ups en PC0, PC1, PC2 y PC4
OUT PORTC, R16     ; Escribir en el registro PORTC

/************* HABILITAR EL TIMER   ********/
CALL    INIT_TMR0
CALL    INIT_TMR1

/************ CONFIGURACIÓN T0 *********/
INIT_TMR0:
    LDI R16, (1<<CS01)//Configurar el prescales en 64 bits
    OUT TCCR0B, R16
    LDI R16, T0VALUE // Valor inicial de TCNT0 para un delay de 2 ms
    OUT TCNT0, R16
    RET
INIT_TMR1:
    LDI R16, HIGH(T1VALUE)
    STS TCNT1H, R16
    LDI R16, LOW(T1VALUE)
    STS TCNT1L, R16

    LDI R16, 0x00
    STS TCCR1A, R16
    LDI R16, (1<<CS01) // configuración para el prescaler de 8
    STS TCCR1B, R16

    RET
```

Este es la configuración de los pines como entradas o salidas según corresponda y el prescaler del CPU en una frecuencia de 1M Hz. También se mandan a llamar las funciones que se encargan de establecer los valores de TCN del timer para que genere la interrupción de tiempo necesaria. En este caso se estableció el prescaler el 8, el Timer 1 con tiempo delay de 500ms y el Timer 0 con un tiempo de delay de 0.2 ms.

Main y Modos

```
MAIN:
    //OUT       PORTB, CONTADOR // mostrar el valor de el contador en el puerto B
    //SBI       PORTD, 2
    CPI     MODO, 0x05
    BREQ    APAGAR_ALARMA_S
    CPI     MODO, 0x00
    BREQ    HORA_S
    CPI     MODO, 0x01
    BREQ    FECHA_S
    CPI     MODO, 0x02
    BREQ    C_HORA_S
    CPI     MODO, 0x03
    BREQ    C_FECHA_S
    CPI     MODO, 0x04
    JMP     C_ALARMA_S
    CPI     MODO, 0x05
    BREQ    APAGAR_ALARMA_S
    RJMP    MAIN
```

Este es el Main del programa, en este se verifica cuál es el valor del registro de Modo, dependiendo del modo se manda a llamar a la función correspondiente.

```
/************** MODOS **********************/
HORA:
    SBRC    ACCION, 0
    CALL    INC_UMIN
    SBI     PORTB, 0
    CBI     PORTB, 1
    SBI     PORTC, 3
    CALL    ACTIVAR_ALARMA
    //CONFIGURACIÓN DE FECHA:

    LDS     CONTADOR_TIEMPO, UMIN  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY1, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    LDS     CONTADOR_TIEMPO, DMIN ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY2, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display2
    LDS     CONTADOR_TIEMPO, UHORAS ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY3, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display3
    LDS     CONTADOR_TIEMPO, DHORAS ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY4, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display4


    RJMP    MAIN
```

Aquí se muestra lo que pasa en el modo de hora. En este se enciende los leds correspondientes que sirven como indicadores de en qué modo nos encontramos. Luego se verifican la bandera de acción. Si el bit 0 de la bandera esta encendida manda

a llamar a la función que corresponde al aumento automático del tiempo. Luego carga los a la variable de display_n el dato correspondiente para cada display. También se llama a la función de activar alarma para que se comparen los valores de configuración de alarma, si llega al valor indicado activar el buzzer.

```
FECHA:

    CBI     PORTB, 0
    SBI     PORTB, 1
    SBI     PORTC, 3
    SBRC    ACCION, 0
    CALL    INC_UMIN
    SBRC    ACCION, 3
    CALL    AUMENTO_DIAS
    SBRC    ACCION, 4
    CALL    AUMENTO_MESES
    //CONFIGURACIÓN DE FECHA:
    LDS     CONTADOR_TIEMPO, MESES  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY1, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    LDS     CONTADOR_TIEMPO, D_MESES ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY2, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display2
    LDS     CONTADOR_TIEMPO, DIAS ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY3, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display3
    LDS     CONTADOR_TIEMPO, D_DIAS ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY4, CONTADOR_TIEMPO ; tomar el valor del registro y guardarlo en el valor que tendrá el display4
    RJMP    MAIN
```

En el modo de fecha se encienden los leds de modo y se verifican los bits de la variable de acción. Si el bit 3 este encendido significa que ya pasaron 24 horas por lo que hay que aumentar la variable de días automáticamente, si la bandera tiene el bit 4 encendido significa que hubo un desbordamiento de días del mes por lo que la variable de unidades de mes debe aumentar una unidad. Se verifica si hay cambio de tiempo con el bit 0 de la bandera y se llama la función ya que el tiempo debe seguir aumentando automáticamente en este modo. Luego se cargan los valores de fecha a cada uno de los displays.

```
C_HORA:
    SBRC    ACCION, 1
    CALL    SUMA
    SBRC    ACCION, 2
    CALL    RESTA
    SBI     PORTB, 0
    CBI     PORTB, 1
    CBI     PORTC, 3
    //Suma y resta de los botones
    LDS     CONTADOR_BOTONES, UD_U_H  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY1, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    LDS     CONTADOR_BOTONES, UD_D_H  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY2, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 2
    LDS     CONTADOR_BOTONES, UD_C_H  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY3, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 3
    LDS     CONTADOR_BOTONES, UD_M_H  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY4, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    //Establecer el inicio en los valores de configuración de hora
    LDS     CONTADOR_TIEMPO, UD_U_H
    STS     UMIN, CONTADOR_TIEMPO
    LDS     CONTADOR_TIEMPO, UD_D_H
    STS     DMIN, CONTADOR_TIEMPO
    LDS     CONTADOR_TIEMPO, UD_C_H
    STS     UHORAS, CONTADOR_TIEMPO
    LDS     CONTADOR_TIEMPO, UD_M_H
```

En el modo de configuración de fecha se verifica so la bandera de acción tiene el bit 1 encendido, lo que verifica si hubo una interrupción de pin change correspondiente al botón de suma. Luego se verifica el bit 2 que corresponde a una acción en el botón de resta. Se muestra la modificación en los displays y posteriormente se carga el valor de las variables de configuración de hora a las variables de hora automática para que inicie el conteo en los valores establecidos.

```
C_FECHA:
    CBI     PORTB, 0
    SBI     PORTB, 1
    CBI     PORTC, 3
    SBRC    ACCION, 1
    CALL    SUMA
    SBRC    ACCION, 2
    CALL    RESTA
    SBRC    ACCION, 0
    CALL    INC_UMIN

    //Suma y resta de los botones
    LDS     CONTADOR_BOTONES, UD_U_F  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY1, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    LDS     CONTADOR_BOTONES, UD_D_F  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY2, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 2
    LDS     CONTADOR_BOTONES, UD_C_F  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY3, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 3
    LDS     CONTADOR_BOTONES, UD_M_F  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY4, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    //CARGAR LOS VALORES DE CONFIGURACIÓN DE FECHA
    LDS     CONTADOR_TIEMPO, UD_U_F
    STS     MESES, CONTADOR_TIEMPO
```

En el modo de fecha se verifican los mismos bits de la bandera que en el modo de configuración de hora. Se cargan los valores de la configuración de hora en los displays y luego se establece el valor de fecha en los valores de la configuración de fecha para que inicie el conteo en los datos correspondientes.

```
C_ALARMA:
    SBRC    ACCION, 0
    CALL    INC_UMIN
    SBRC    ACCION, 1
    CALL    SUMA
    SBRC    ACCION, 2
    CALL    RESTA
    CBI     PORTB, 0
    CBI     PORTB, 1
    SBI     PORTC, 3
    //Suma y resta de los botones
    LDS     CONTADOR_BOTONES, UD_U_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY1, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    LDS     CONTADOR_BOTONES, UD_D_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY2, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 2
    LDS     CONTADOR_BOTONES, UD_C_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY3, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 3
    LDS     CONTADOR_BOTONES, UD_M_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY4, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    RJMP    MAIN
```

Se verifican las mismas banderas que en los modos de configuración anteriores. Se carga a los displays el valor de configuración de alarma establecido.

```
APAGAR_ALARMA:
    LDS     CONTADOR_BOTONES, UD_U_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY1, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    LDS     CONTADOR_BOTONES, UD_D_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY2, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 2
    LDS     CONTADOR_BOTONES, UD_C_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY3, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 3
    LDS     CONTADOR_BOTONES, UD_M_A  ; Tomar el valor de unidades y guardarlo en el registro
    STS     DISPLAY4, CONTADOR_BOTONES ; tomar el valor del registro y guardarlo en el valor que tendrá el display 1
    SBRC    ACCION, 0
    CALL    INC_UMIN
    CBI     PORTB, 0
    CBI     PORTB, 1
    CBI     PORTC, 3
    CBI     PORTC, 5
    RJMP    MAIN
```

Se carga el valor de configuración alarma a los displays y se apaga el pin del buzzer para apagar el sonido de la alarma.

Rutina de interrupción de Pin change y subrutinas para modos de configuración (aumento y decremento)

```
ISR_PCINT1:
    PUSH    R16
    IN      R16, SREG
    PUSH    R16
    //Para el botón de modos
    SBIS    PINC, PC2                   //Leer si el botón de cambio de modo está en set, si lo está saltar la siguiente línea
    JMP     BOTONMODO
    SBIS    PINC, PC0
    JMP     BOTON_SUMA
    SBIS    PINC, PC1
    JMP     BOTON_RESTA
    SBIS    PINC, PC4
    JMP     UNIDADES_DECENAS
    JMP     F_ISR
BOTONMODO:
    INC     MODO
    LDI     R16, MODOS
    CPSE    MODO, R16 //Saltar si son iguales
    JMP     F_ISR
    CLR     MODO ; Si el modo se pasó del límite limpiarlo y comenzar el 0
    JMP     F_ISR
BOTON_SUMA:
    LDI     R16, 0b00000010
    EOR     ACCION, R16
    JMP     F_ISR
BOTON_RESTA:
    LDI     R16, 0b00000100
    EOR     ACCION, R16
    JMP     F_ISR

UNIDADES_DECENAS:
    LDS     CONTADOR_BOTONES, U_D
    INC     CONTADOR_BOTONES
    STS     U_D, CONTADOR_BOTONES
    LDI     R16, 0x02
```

Esta es la rutina de interrupción del timer, cada vez que haya un cambio en los estados verifica si un bit especifico del Pin C está en set, si está en set significa que no se presionó el botón (debido a la lógica del pull up), si está en set se salta la línea que llama a la función correspondiente. Si se presionó el pin 2 es el botón de cambio de modo, el pin 0 del de suma, el 1 el de resta y el 4 el que permite seleccionar que para de diplays se quiere modificar. Luego de parar a las funciones estas encienden el bit de la bandera que corresponda para que en el modo se pueda llamar a la función necesaria. Luego de encender las banderas se sale de la rutina de interrupción, con las banderas se puede evitar que la rutina de interrupción sea demasiado larga.

```
SUMA:
    LDI     R16, 0b00000010   ; Cargar el valor en R16
    EOR     ACCION, R16       ; Alternar el bit correspondiente en ACCION

    ; Comprobar si MODO == 0x02
    CPI     MODO, 0x02
    BRNE    VERIFICAR_FECHA1   ; Si no es igual, verificar la siguiente condición
    JMP     SUMA_HORA         ; Si es igual, saltar a SUMA_HORA

VERIFICAR_FECHA1:
    ; Comprobar si MODO == 0x03
    CPI     MODO, 0x03
    BRNE    VERIFICAR_ALARMA   ; Si no es igual, saltar a retorno largo
    JMP     SUMA_FECHA         ; Si es igual, saltar a SUMA_FECHA
VERIFICAR_ALARMA:
    CPI     MODO, 0x04
    BRNE    LLAMAR_RETORNO   ; Si no es igual, saltar a retorno largo
    JMP     SUMA_ALARMA        ; Si es igual, saltar a SUMA_ALARMA

LLAMAR_RETORNO:
    JMP     RETORNO_BOTON     ; Usar JMP para saltar a cualquier parte del código
/*************************************************************SUBRUTINAS PARA EL BOTÓN DE SUMA EN MODO HORA ****************************/
SUMA_HORA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00 ;si el botón de suma fue el que se presionó comparar en que modo se está
    BREQ    SUMA_HORA_UNIDADES
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    SUMA_HORA_DECENAS
    JMP     RETORNO_BOTON
SUMA_HORA_UNIDADES:
    //Para solo modificar en un solo modo
```

Acá se verifica en que modo nos encontramos en este momento para llamar la la función correspondiente, esto permite modificar únicamente la variable designada para el modo.

```
/*************************************************************SUBRUTINAS PARA EL BOTON DE SUMA EN MODO HORA *******
SUMA_HORA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00 ;si el botón de suma fue el que se presionó comparar en que modo se está
    BREQ    SUMA_HORA_UNIDADES
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    SUMA_HORA_DECENAS
    JMP     RETORNO_BOTON
SUMA_HORA_UNIDADES:
    //Para solo modificar en un solo modo
    LDS     CONTADOR_BOTONES, UD_U_H
    //Ahora se le suma el contador a las unidades de los min
    CPI     CONTADOR_BOTONES,MAX_UNI
    BREQ    OFUC
    INC     CONTADOR_BOTONES        ; incrementa la variable
    STS     UD_U_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
OFUC:
    LDI     CONTADOR_BOTONES, 0x00
    STS     UD_U_H, CONTADOR_BOTONES ; Limpiar las unidades
    LDS     CONTADOR_BOTONES, UD_D_H
    INC     CONTADOR_BOTONES
    STS     UD_D_H, CONTADOR_BOTONES
    LDI     R16, 0x06
    CPSE    CONTADOR_BOTONES, R16 ; Saltar la siguiente linea si son iguales
    JMP     RETORNO_BOTON
    CLR     CONTADOR_BOTONES
    STS     UD_D_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON

SUMA_HORA_DECENAS:
    LDS     CONTADOR_BOTONES, UD_M_H     ; Cargar decenas de horas
    CPI     CONTADOR_BOTONES, 0x02        ; Verificar si DHORAS == 2
```

En el modo de hora se verifica cuál de los dos pares de displays se quiere modificar. Si es unidades se hace la lógica de incremento con overflow, si las unidades son 9, pasar

a la siguiente función que establece las unidades de minutos en 0 y suma uno a las decenas de minuto.

```
SUMA_HORA_DECENAS:
    LDS     CONTADOR_BOTONES, UD_M_H     ; Cargar decenas de horas
    CPI     CONTADOR_BOTONES, 0x02        ; Verificar si DHORAS == 2
    BRNE    OFT_C         ; Si es 2, verificar si UHORAS == 4 (24 horas)
    LDS     CONTADOR_BOTONES, UD_C_H
    CPI     CONTADOR_BOTONES,  0x03
    BRNE    MAX_FIN_DIA_C ; MIENTRAS NO SEA 4 IR A LA FUNCION
    CLR     CONTADOR_BOTONES
    STS     UD_U_H, CONTADOR_BOTONES
    STS     UD_D_H, CONTADOR_BOTONES
    STS     UD_C_H, CONTADOR_BOTONES
    STS     UD_M_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON

OFT_C:
    // Incrementar decenas
    LDS     CONTADOR_BOTONES, UD_C_H
    //Ahora se le suma el contador a las unidades de los min
    CPI     CONTADOR_BOTONES, MAX_UNI
    BREQ    MAX_D_TIEMPO_C
    INC     CONTADOR_BOTONES        ; incrementa la variable
    STS     UD_C_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
MAX_D_TIEMPO_C:
    LDI     CONTADOR_BOTONES, 0x00
    STS     UD_C_H, CONTADOR_BOTONES ;LIMPIAR UNIDADES
    LDS     CONTADOR_BOTONES,  UD_M_H
    INC     CONTADOR_BOTONES
    STS     UD_M_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
MAX_FIN_DIA_C:
    LDS     CONTADOR_BOTONES, UD_C_H
    INC     CONTADOR BOTONES
```

Esta es la función para modificar la hora del modo de configuración de hora. En este se debe establecer la lógica para aumentar las horas con un overflow de 24 horas. En primer lugar, verifica si las decenas de hora son 2, si es este valor verificar si las unidades son 3. Si las unidades son 3 continuar para establecer todas las variables de hora en 0. Si en el inicio de las comparaciones las decenas no son dos, saltar a una función que verifica si las unidades son 9, si lo son saltar a una función que establece las unidades en 0 y suma 1 a las decenas; si no llegaron a 9 incrementar normalmente las unidades. Si las decenas son 2 pero las unidades no son 3, saltar a la función que incrementa las unidades.

```
SUMA_FECHA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00 ;si el botón de suma fue el que se presionó comparar en que modo se está
    BREQ    SUMA_FECHA_UNIDADES_C
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    SUMA_FECHA_DECENAS
    JMP     RETORNO_BOTON
SUMA_FECHA_UNIDADES_C:
    //Para incrementar la variable que usaremos en días
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    CPI     CONTADOR_BOTONES, 12
    BREQ    CONTEO_MESES_CLR
    INC     CONTADOR_BOTONES
    STS     CONTEO_MESES, CONTADOR_BOTONES
    JMP     SUMA_FECHA_UNIDADES
CONTEO_MESES_CLR:
    LDI     CONTADOR_BOTONES, 0x00
    STS     CONTEO_MESES, CONTADOR_BOTONES
    JMP     SUMA_FECHA_UNIDADES
SUMA_FECHA_UNIDADES:
    //Para solo modificar en un solo modo
    LDS     CONTADOR_BOTONES, UD_D_F
    CPI     CONTADOR_BOTONES, 0x01
    BRNE    OFU_MESES
    LDS     CONTADOR_BOTONES, UD_U_F
    CPI     CONTADOR_BOTONES, 0x02
    BRNE    MAX_FIN
    LDI     CONTADOR_BOTONES,0x1
    STS     UD_U_F, CONTADOR_BOTONES
    CLR     CONTADOR_BOTONES
    STS     UD_D_F, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
```

Esta es la función encargada de sumar en configuración de fecha. En esta en primer lugar se suma una variable que aumenta cada vez que aumenta la configuración del mes, esta se utilizará para realizar el overflow en el aumento de días. En la siguiente función se hace el aumento de meses con la misma lógica que aumento de días, pero se establece el límite de decenas en 1 y el de unidades en 2.

```
SUMA_FECHA_DECENAS:
    //CARGAR EL VALOR DE LIMITE PARA UNIDADES
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    LDI     ZH, HIGH(TABLA_DIAS_U<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA_DIAS_U<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, CONTADOR_BOTONES //Sumar la posición del contador de meses
    LPM     R16, Z
    STS     LIMITE_U, R16
    //CARGAR EL VALOR DEL LIMITE PARA LAS DECENAS
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    LDI     ZH, HIGH(TABLA_DIAS_D<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA_DIAS_D<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, CONTADOR_BOTONES //Sumar la posición del contador de meses
    LPM     R16, Z
    STS     LIMITE_D, R16
    //LÓGICA DE COMPARACIÓN
    LDS     CONTADOR_BOTONES, UD_M_F
    LDS     R16, LIMITE_D
    CP      CONTADOR_BOTONES, R16 //Comparar con el límite de las decenas
    BRNE    MES_N
    LDS     R16, LIMITE_U
    LDS     CONTADOR_BOTONES, UD_C_F
    CP      CONTADOR_BOTONES, R16 //COMPARAR CON EL LIMITE DE UNIDADES
    BRNE    MAX_FIN_MESES
    LDI     CONTADOR_BOTONES, 0x01
    STS     UD_C_F, CONTADOR_BOTONES
    CLR     CONTADOR_BOTONES
    STS     UD_M_F, CONTADOR_BOTONES
    JMP     RETORNO_BOTON


MES_N:
    // Incrementa las decenas y verificar si no ha exedido unidades
    LDS     CONTADOR_BOTONES, UD_C_F
    //Ahora se le suma el contador a las unidades de los min
    CPI     CONTADOR_BOTONES,MAX_UNI //verifica si no es 9
    BREQ    MAX_DM
    INC     CONTADOR_BOTONES        ; incrementa la variable
    STS     UD_C_F, CONTADOR_BOTONES
    JMP     RETORNO_BOTON

MAX_DM:
    LDI     CONTADOR_BOTONES, 0x00
    STS     UD_C_F, CONTADOR_BOTONES ;LIMPIAR UNIDADES
    LDS     CONTADOR_BOTONES, UD_M_F
    INC     CONTADOR_BOTONES ; SUMAR EN DECENAS
    STS     UD_M_F, CONTADOR_BOTONES
    JMP     RETORNO_BOTON

MAX_FIN_MESES:
    LDS     CONTADOR_BOTONES, UD_C_F
    INC     CONTADOR_BOTONES
    STS     UD_C_F, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
```

Los meses tiene diferentes días y estos no tiene un patrón específico, por esta razón se crearon las tablas, una para unidades y otra para decenas que tienen los números que servirán como límite de comparación en el overflow. En la suma de las decenas (días) se establecen los límites sumando al puntero Z el valor de la variable de conteo de modificación de meses establecida anteriormente, luego se sigue la misma lógica explicada en los modos de suma anteriores.

```
/*************************************************************SUBRUTINAS PARA EL BOTÓN DE SUMA EN MODO CONFIGURACIÓN DE ALARMA ****************************/
SUMA_ALARMA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00 ;si el botón de suma fue el que se presionó comparar en que modo se está
    BREQ    SUMA_ALARMA_UNIDADES
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    SUMA_ALARMA_DECENAS
    JMP     RETORNO_BOTON

SUMA_ALARMA_UNIDADES:
    //Para solo modificar en un solo modo
    LDS     CONTADOR_BOTONES, UD_U_A
    //Ahora se le suma el contador a las unidades de los min
    CPI     CONTADOR_BOTONES,MAX_UNI
    BREQ    OFUC_ALARMA
    INC     CONTADOR_BOTONES        ; incrementa la variable
    STS     UD_U_A, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
OFUC_ALARMA:
    LDI     CONTADOR_BOTONES, 0x00
    STS     UD_U_A, CONTADOR_BOTONES ; Limpiar las unidades
    LDS     CONTADOR_BOTONES, UD_D_A
    INC     CONTADOR_BOTONES
    STS     UD_D_A, CONTADOR_BOTONES
    LDI     R16, 0x06
    CPSE    CONTADOR_BOTONES, R16 ; Saltar la siguiente línea si son iguales
    JMP     RETORNO_BOTON
    CLR     CONTADOR_BOTONES
    STS     UD_D_A, CONTADOR_BOTONES
```

En suma de configuración de alarma se sigue la misma lógica que en configuración de hora solo se cambian las variables utilizadas.

```
RESTA:
    LDI     R16, 0b00000100      ; Cargar el valor en R16
    EOR     ACCION, R16          ; Alternar el bit correspondiente en ACCION
    JMP     VERIFICAR_MODO       ; Saltar a la lógica de comparación

VERIFICAR_MODO:
    ; Comprobar si MODO == 0x02
    CPI     MODO, 0x02
    BRNE    VERIFICAR_FECHA      ; Si no es igual, verificar la siguiente condición
    JMP     RESTA_HORA           ; Si es igual, saltar a RESTA_HORA

VERIFICAR_FECHA:
    ; Comprobar si MODO == 0x03
    CPI     MODO, 0x03
    BRNE    VERIFICAR_ALARMA_R       ; Si no es igual, regresar
    JMP     RESTA_FECHA          ; Si es igual, saltar a RESTA_FECHA

VERIFICAR_ALARMA_R:
    ; Comprobar si MODO == 0x04
    CPI     MODO, 0x04
    BRNE    LLAMAR_R        ; Si no es igual, regresar
    JMP     RESTA_ALARMA          ; Si es igual, saltar a RESTA_FECHA
LLAMAR_R:
    JMP     RETORNO_BOTON
/*************************************************************SUBRUTINAS PARA EL BOTÓN DE RESTA EN MODO CONFIGURACIÓN DE HORA ****************************/
RESTA_HORA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00 ;si el botón de suma fue el que se presionó comparar en que modo se está
    BREQ    RESTA_HORA_UNIDADES
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    RESTA_HORA_DECENAS
    JMP     RETORNO_BOTON
RESTA_HORA_UNIDADES:
```

Al igual que en suma en resta se verifica cual es el modo que se quiere configurar, luego salta a la función correspondiente.

```
RESTA_HORA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00 ;si el botón de suma fue el que se presionó comparar en que modo se está
    BREQ    RESTA_HORA_UNIDADES
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    RESTA_HORA_DECENAS
    JMP     RETORNO_BOTON
RESTA_HORA_UNIDADES:
    //Para solo modificar en un solo modo
    LDS     CONTADOR_BOTONES, UD_U_H
    //Ahora se le RESTA el contador a las unidades de los min
    CPI     CONTADOR_BOTONES,0x00
    BREQ    OUFU
    DEC     CONTADOR_BOTONES        ; incrementa la variable
    STS     UD_U_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
OUFU:
    LDI     CONTADOR_BOTONES, 0x09
    STS     UD_U_H, CONTADOR_BOTONES ; Limpiar las unidades
    LDS     CONTADOR_BOTONES, UD_D_H
    CPI     CONTADOR_BOTONES, 0x00
    BREQ    OUFU2
    DEC     CONTADOR_BOTONES
    STS     UD_D_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
OUFU2:
    LDS     CONTADOR_BOTONES, UD_D_H
    LDI     CONTADOR_BOTONES, 0x05
    STS     UD_D_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON

RESTA_HORA_DECENAS:
```

En resta hora se verifica si se modifican los minutos o las horas. Si se modifican los minutos salta a resta unidad hora, en esta función se hace el decremento de las unidades de las variables. Si esta llega a 0 saltar a otra función que verifica si las decenas son 0, si las decenas no son 0 se establecen las unidades en 9 y se decrementan las decenas; si las decenas eran 0, saltar a una función que establezca las decenas en 5 y las unidades en 9.

```
RESTA_HORA_DECENAS:
    LDS     CONTADOR_BOTONES, UD_M_H
    CPI     CONTADOR_BOTONES, 0x00
    BRNE    UFU_HORAS  // Salta si no es el primer caso
    LDS     CONTADOR_BOTONES, UD_C_H
    CPI     CONTADOR_BOTONES, 0x00
    BRNE    DECREMENTAR_UNI         // si es 0 saltar
    // si los dos son 0 establecer el contador en 24
    LDI     CONTADOR_BOTONES, 0x03
    STS     UD_C_H, CONTADOR_BOTONES
    LDI     CONTADOR_BOTONES, 0x02
    STS     UD_M_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
UFU_HORAS:
    LDS     CONTADOR_BOTONES, UD_C_H // Si las decenas no son 0 d
    CPI     CONTADOR_BOTONES, 0x00 // comparar con 0
    BREQ    DECREMENTAR_DEC // si es 0 saltar
    DEC     CONTADOR_BOTONES // si no es o decrementar las unidades normalmente.
    STS     UD_C_H, CONTADOR_BOTONES
    JMP     RETORNO_BOTON

DECREMENTAR_DEC:
    // Si las unidades son 0, decrementa las decenas
    LDS     CONTADOR_BOTONES, UD_M_H
    DEC     CONTADOR_BOTONES
    STS     UD_M_H, CONTADOR_BOTONES
    // establecer las unidades en 9
    LDI     CONTADOR_BOTONES, 0x09
    STS     UD_C_H, CONTADOR_BOTONES
```

En la resta decenas se verifica si las decenas son 0, si son 0 se verifica si las unidades también son 0, si los son establece las unidades en 3 y las decenas en 2. Si las decenas en la primera comparación no son 0 salta a una función que verifica si las unidades son

0, si lo son saltar a la función de decrementar decenas, si las unidades no eran 0 decrementarlas con normalidad. En la función de decrementar decenas, disminuye en uno las decenas y establece las unidades en 9. Si las decenas eran 0 pero las unidades no eran 0 se salta a una función que decrementa las unidades con normalidad.

```
/*************************************************************SUBRUTINAS PARA EL BOTÓN DE RESTA EN MODO FECHA UNIDADES ***
RESTA_FECHA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00
    BREQ    RESTA_FECHA_UNIDADES_C
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    RESTA_FECHA_DECENAS
    JMP     RETORNO_BOTON

RESTA_FECHA_UNIDADES_C:
    ; Para decrementar los meses
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    CPI     CONTADOR_BOTONES, 0x00
    BREQ    CONTEO_MESES_CLR_R       ; Si es 0, reinicia a 12
    DEC     CONTADOR_BOTONES
    STS     CONTEO_MESES, CONTADOR_BOTONES
    JMP     RESTA_FECHA_UNIDADES

CONTEO_MESES_CLR_R:
    ; Si llega a 0, reiniciar a 12
    LDI     CONTADOR_BOTONES, 0x12
    STS     CONTEO_MESES, CONTADOR_BOTONES
    JMP     RETORNO_BOTON

RESTA_FECHA_UNIDADES:
    ; Para restar en las unidades de los meses
    LDS     CONTADOR_BOTONES, UD_D_F
    CPI     CONTADOR_BOTONES, 0x00   ; saltar y seguir decenas sean 0
    BRNE    UFU_MESES_R              ; Si no, sigue con las unidades

    ; Si decenas es 0, verificar unidades
    LDS     CONTADOR_BOTONES, UD_U_F
    CPI     CONTADOR_BOTONES, 0x01   ; si las unidades son 1 saltar
    BREQ    RESTA A 12               ; Si sí, reiniciar a 12
```

En resta de fecha unidades se sigue la misma lógica que en el decremento de horas, solo que los límites de comparación en decenas son de 0 y en unidades en de 1. También se nota que si se resta las unidades de fecha (los meses) el contador que se utilizará en el puntero también disminuye.

```
RESTA_FECHA_DECENAS:
//CARGAR EL VALOR DE LIMITE PARA UNIDADES
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    LDI     ZH, HIGH(TABLA_DIAS_U<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA_DIAS_U<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, CONTADOR_BOTONES //Sumar la posición del contador de meses
    LPM     R16, Z
    STS     LIMITE_U, R16
    //CARGAR EL VALOR DEL LIMITE PARA LAS DECENAS
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    LDI     ZH, HIGH(TABLA_DIAS_D<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA_DIAS_D<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, CONTADOR_BOTONES //Sumar la posición del contador de meses
    LPM     R16, Z
    STS     LIMITE_D, R16
    //LÓGICA DE COMPARACIÓN
    LDS     CONTADOR_BOTONES, UD_M_F
    CPI     CONTADOR_BOTONES, 0x00
    BRNE    UFU_MESES_D   // Salta si no es el primer caso
    LDS     CONTADOR_BOTONES, UD_C_F
    CPI     CONTADOR_BOTONES, 0x01
    BRNE    DECREMENTAR_UNI_D         // si es 0 saltar
    // si los dos son 0 establecer el contador en en los limites de la tabla
    LDS     CONTADOR_BOTONES, LIMITE_U
    STS     UD_C_F, CONTADOR_BOTONES
    LDS     CONTADOR_BOTONES, LIMITE_D
    STS     UD_M_F, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
UFU_MESES_D:
    LDS     CONTADOR_BOTONES, UD_C_F // Si las decenas no son 0 d
    CPI     CONTADOR_BOTONES, 0x00 // comparar con 0
    BREQ    DECREMENTAR_DEC_D // si es 0 saltar
```

Se sigue la misma lógica de underflow que en las configuraciones paradas, y al igual que en suma de días se utilizan las tablas para establecer los límites de comparación de las decenas y unidades.

```
    JMP     RETORNO_BOTON
/************************************************************SUBRUTINAS PARA EL BOTÓN DE RESTA EN MODO CONFIGURACIÓN DE ALARMA ****************************/
RESTA_ALARMA:
    LDS     CONTADOR_BOTONES, U_D
    CPI     CONTADOR_BOTONES, 0x00 ;si el botón de suma fue el que se presionó comparar en que modo se está
    BREQ    RESTA_ALARMA_UNIDADES
    CPI     CONTADOR_BOTONES, 0x01
    BREQ    RESTA_ALARMA_DECENAS
    JMP     RETORNO_BOTON
RESTA_ALARMA_UNIDADES:
    //Para solo modificar en un solo modo
    LDS     CONTADOR_BOTONES, UD_U_A
    //Ahora se le RESTA el contador a las unidades de los min
    CPI     CONTADOR_BOTONES,0x00
    BREQ    OUFU_A
    DEC     CONTADOR_BOTONES        ; incrementa la variable
    STS     UD_U_A, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
OUFU_A:
    LDI     CONTADOR_BOTONES, 0x09
    STS     UD_U_A, CONTADOR_BOTONES ; Limpiar las unidades
    LDS     CONTADOR_BOTONES, UD_D_A
    CPI     CONTADOR_BOTONES, 0x00
    BREQ    OUFU2_A
    DEC     CONTADOR_BOTONES
    STS     UD_D_A, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
OUFU2_A:
    LDS     CONTADOR_BOTONES, UD_D_A
    LDI     CONTADOR_BOTONES, 0x05
    STS     UD_D_A, CONTADOR_BOTONES
    JMP     RETORNO_BOTON
```

Se utiliza la misma lógica que en resta en configuración de horas, solo se modifican las variables utilizadas.

Rutina de interrupción del Timer 0 y Mux

```
/************************************ INTERRUPCIONES DEL T0*****************************************/
TMR0_ISR:
    PUSH    R16
    IN      R16, SREG
    PUSH    R16
    //Establecer los leds en apagado inicialmente
    CBI     PORTB, 2
    CBI     PORTB, 3
    CBI     PORTB, 4
    CBI     PORTB, 5
    INC     CONTADORT0
    //VERIFICAR QUE NO HAYA EXEDIDO EL LÍMITE
    LDI     R16, MAXT0
    CPSE    CONTADORT0, R16 //Saltar si son iguales
    JMP     MUX
    LDI     CONTADORT0, 0x00
    JMP     FIN_T0
MUX:
    CPI     CONTADORT0, 0x01          ; PB2
    BREQ    DISPLAY_1
    CPI     CONTADORT0, 0x02          ; PB3
    BREQ    DISPLAY_2
    CPI     CONTADORT0, 0x03          ; PB4
    BREQ    DISPLAY_3
    CPI     CONTADORT0, 0x04          ; PB5
    BREQ    DISPLAY_4
    JMP     FIN_T0
```

El timer 0 se utilizó para hacer el multiplexeo de los displays. En esta se tiene un contador que cada vez que pasa por la interrupción aumenta. Si el contador excede el límite de displays se reinicia. En la función de mux se evalúa cual es el valor del contador y se va la función para encender el pin del transistor correspondiente.

```
    JMP     FIN_T0
DISPLAY_1:
    //ENCENDER SOLO EL TRANSISTOR NECESARIO
    SBI     PORTB, 2
    LDS     DISPLAYS, DISPLAY1 ; El registro de display tiene la salida de display 1 según el modo
    //SOLO PARA PROBAR QUE EL MUX FUNCIONE
    LDI     ZH, HIGH(TABLA<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, DISPLAYS
    LPM     S_DISPLAY, Z
    OUT     PORTD, S_DISPLAY
    JMP     FIN_T0
DISPLAY_2:
    //ENCENDER SOLO EL TRANSISTOR NECESARIO
    SBI     PORTB, 3
    LDS     DISPLAYS, DISPLAY2
    //SOLO PARA PROBAR QUE EL MUX FUNCIONE
    LDI     ZH, HIGH(TABLA<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, DISPLAYS
    LPM     S_DISPLAY, Z
    OUT     PORTD, S_DISPLAY
    JMP     FIN_T0
DISPLAY_3:
    //ENCENDER SOLO EL TRANSISTOR NECESARIO
    SBI     PORTB, 4
    LDS     DISPLAYS, DISPLAY3
    //SOLO PARA PROBAR QUE EL MUX FUNCIONE
    LDI     ZH, HIGH(TABLA<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, DISPLAYS
    LPM     S_DISPLAY, Z
    OUT     PORTD, S_DISPLAY
    JMP     FIN_T0
DISPLAY_4:
```

En estas funciones se establece el puntero en la primera función de la tabla y se le suma el registro de displays, este registro contiene el valor establecido en los modos. Y luego de pasar por la función de display se termina la rutina de interrupción del timer.

Rutina de interrupción del Timer 1, aumento de tiempo y aumento automático de días.

```
/*************** INTERRUPCIONES DEL T1************/
TMR1_ISR:
    PUSH    R16
    IN      R16, SREG
    PUSH    R16
    LDI     R16, 0b00000100
    //PAPADEO DE LOS LEDS
    EOR     S_DISPLAY, R16
    OUT     PORTD, S_DISPLAY

    //Conteo de Tiempo
    LDS     CONTADOR_TIEMPO, MINUTO
    INC     CONTADOR_TIEMPO
    STS     MINUTO, CONTADOR_TIEMPO

    ; Verificar si ya pasó el tiempo necesario
    CPI     CONTADOR_TIEMPO, CICLO
    BRNE    FIN_TMR1  ; Si no ha llegado salir
    CLR     CONTADOR_TIEMPO
    STS     MINUTO, CONTADOR_TIEMPO
    //Resetear la bandera
    LDI     R16, 0x01
    EOR     ACCION, R16
    JMP     FIN_TMR1
FIN_TMR1:
    POP     R16
    OUT     SREG, R16
    POP     R16
    RETI
```

Esta es la rutina de interrupción del timer 1, lo primero que se hace es verificar si el valor del contador no ha excedido el número de ciclos necesarios para que haya pasado 1 min. Si ya se excedió ese tiempo enciende la bandera para el incremento de minutos y termina la interrupción. También se debe mencionar que hay en EOR para que alterne entre encendido y apagado los leds cada 500 ms.

```
INC_UMIN:
    //Resetear la bandera
    LDI     R16, 0x01
    EOR     ACCION, R16
    // SUMAR A UMIN
    LDS     CONTADOR_TIEMPO, UMIN
    CPI     CONTADOR_TIEMPO, MAX_UNI ; COMPARAR PARA VER SI SUPERO UNIDADES
    BREQ    OFU
    INC     CONTADOR_TIEMPO
    STS     UMIN, CONTADOR_TIEMPO
    JMP     RETORNOH

OFU:
    LDI     CONTADOR_TIEMPO, 0x00        ; Reiniciar unidades de minutos
    STS     UMIN, CONTADOR_TIEMPO
    LDS     CONTADOR_TIEMPO, DMIN        ; Cargar decenas de minutos
    CPI     CONTADOR_TIEMPO, MAX_DEC     ; Verificar si DMIN == 5
    BREQ    OFDH                         ; Si es 5, reiniciar decenas y unidades
    INC     CONTADOR_TIEMPO              ; Si no, incrementar DMIN
    STS     DMIN, CONTADOR_TIEMPO
    JMP     RETORNOH
OFDH:
    LDI     CONTADOR_TIEMPO, 0x00        ; Reiniciar unidades de minutos
    STS     UMIN, CONTADOR_TIEMPO
    STS     DMIN, CONTADOR_TIEMPO        ; Reiniciar decenas de minutos
    LDS     CONTADOR_TIEMPO, DHORAS      ; Cargar decenas de horas
    CPI     CONTADOR_TIEMPO, 0x02        ; Verificar si DHORAS == 2
    BRNE    OFT       ; Si es 2, verificar si UHORAS == 4 (24 horas)
    LDS     CONTADOR_TIEMPO, UHORAS
    CPI     CONTADOR_TIEMPO,  0x03
    BRNE    MAX_FIN_DIA ; MIENTRAS NO SEA 4 IR A LA FUNCION
    CLR     CONTADOR_TIEMPO
    STS     UMIN, CONTADOR_TIEMPO
    STS     DMIN, CONTADOR_TIEMPO
    STS     UHORAS, CONTADOR_TIEMPO
    STS     UHORAS, CONTADOR_TIEMPO
    STS     DHORAS, CONTADOR_TIEMPO
    LDI     R16, 0b00001000
    EOR     ACCION, R16
    JMP     RETORNOH
```

Esta es la función de aumento de tiempo que se llama en los modos, esta aumenta el tiempo con la misma lógica de incremento y overflow utilizada en los modos de configuración. Se debe observar que una vez cumplidas las 24 horas se enciende el bit de la bandera que corresponde a un aumento de días.

```
AUMENTO_DIAS:
    LDI     R16, 0b00001000
    EOR     ACCION, R16
    //LÓGICA DEL AUMENTO
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    LDI     ZH, HIGH(TABLA_DIAS_U<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA_DIAS_U<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, CONTADOR_BOTONES //Sumar la posición del contador de meses
    LPM     R16, Z
    STS     LIMITE_U, R16
    //CARGAR EL VALOR DEL LIMITE PARA LAS DECENAS
    LDS     CONTADOR_BOTONES, CONTEO_MESES
    LDI     ZH, HIGH(TABLA_DIAS_D<<1)  // Carga la parte alta de la dirección de tabla en ZH
    LDI     ZL, LOW(TABLA_DIAS_D<<1)   // Carga la parte baja de la dirección de la tabla en ZL
    ADD     ZL, CONTADOR_BOTONES //Sumar la posición del contador de meses
    LPM     R16, Z
    STS     LIMITE_D, R16
    //LÓGICA DE COMPARACIÓN
    LDS     CONTADOR_BOTONES, D_DIAS
    LDS     R16, LIMITE_D
    CP      CONTADOR_BOTONES, R16 //Comparar con el límite de las decenas
    BRNE    MES_A
    LDS     R16, LIMITE_U
    LDS     CONTADOR_BOTONES, DIAS
    CP      CONTADOR_BOTONES, R16 //COMPARAR CON EL LIMITE DE UNIDADES
    BRNE    MAX_FIN_MESES_A
    LDI     CONTADOR_BOTONES, 0x01
    STS     DIAS, CONTADOR_BOTONES
    CLR     CONTADOR_BOTONES
    STS     D_DIAS, CONTADOR_BOTONES
    LDI     R16, 0b00010000
    EOR     ACCION, R16
    JMP     RETORNOH
```

Una vez de mande a llamar la función de aumento de días se utiliza la misma lógica de incremento de días y overflow que en el modo de configuración de fecha. Una vez haya overflow total de los días del mes se enciende el bit de la bandera correspondiente al aumento de meses. En esta función se modifican las variables del aumento automático de fecha.

```
AUMENTO_MESES:
    LDI     R16, 0b00010000
    EOR     ACCION, R16
    //LÓGICA DE AUMENTO DE MESES
    LDS     CONTADOR_BOTONES, D_MESES
    CPI     CONTADOR_BOTONES, 0x01
    BRNE    OFU_MESES_A
    LDS     CONTADOR_BOTONES, MESES
    CPI     CONTADOR_BOTONES, 0x02
    BRNE    MAX_FIN_A
    LDI     CONTADOR_BOTONES,0x1
    STS     MESES, CONTADOR_BOTONES
    CLR     CONTADOR_BOTONES
    STS     D_MESES, CONTADOR_BOTONES
    JMP     RETORNOH

OFU_MESES_A:
    // Incrementar decenas Y VERIFICA SI NO ES 1
    LDS     CONTADOR_BOTONES, MESES
    //Ahora se le suma el contador a las unidades de los min
    CPI     CONTADOR_BOTONES,MAX_UNI
    BREQ    MAX_D_A
    INC     CONTADOR_BOTONES        ; incrementa la variable
    STS     MESES, CONTADOR_BOTONES
    JMP     RETORNOH


MAX_FIN_A:
    LDS     CONTADOR_BOTONES, MESES
    INC     CONTADOR_BOTONES
    STS     MESES, CONTADOR_BOTONES
    JMP     RETORNOH
MAX_D_A:
    LDI     CONTADOR_BOTONES, 0x00
```

El aumento de meses utiliza la misma lógica que en la configuración de fecha, solo que se modifican las variables utilizadas en el aumento automático de fecha.

Activar la alarma

```
ACTIVAR_ALARMA:
    LDS     R16, DHORAS
    LDS     ALARMA_V, UD_M_A
    CP      R16, ALARMA_V
    BREQ    VERIFICAR_UHORAS
    JMP     RETORNO_ALARMA
VERIFICAR_UHORAS:
    LDS     R16, UHORAS
    LDS     ALARMA_V, UD_C_A
    CP      R16, ALARMA_V
    BREQ    VERIFICAR_DMIN
    JMP     RETORNO_ALARMA
VERIFICAR_DMIN:
    LDS     R16, DMIN
    LDS     ALARMA_V, UD_D_A
    CP      R16, ALARMA_V
    BREQ    VERIFICAR_UMIN
    JMP     RETORNO_ALARMA
VERIFICAR_UMIN:
    LDS     R16, UMIN
    LDS     ALARMA_V, UD_U_A
    CP      R16, ALARMA_V
    BREQ    SONAR_ALARMA
    JMP     RETORNO_ALARMA
SONAR_ALARMA:
    SBI     PORTC, 5
    JMP     RETORNO_ALARMA
RETORNO_ALARMA:
    RET
// Tabla para 7 segmentos
TABLA: .DB 0x7B, 0x0A, 0xB3, 0x9B, 0xCA, 0xD9, 0xF9, 0x0B, 0xFB, 0xDB, 0xEB, 0xF8, 0x71, 0xB4, 0xF1, 0xE1
TABLA_DIAS_U: .DB 0x01, 0x08, 0x01, 0x00, 0x01, 0x00, 0x01, 0x01, 0x00, 0x01, 0x00, 0x01
TABLA_DIAS_D: .DB 0x03, 0x02, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03, 0x03
```

Este es la función que se llama en modo hora automática, esta compara las desde las decenas de hora hasta las unidades de minuto, si todos los valores coinciden se llega a la función de sonar alarma que activa el Pin 5 del puerto C al que está conectado del buzzer, haciendo que suene la alarma.

En la parte inferior se pueden ver las tablas utilizadas para mostrar los valores en los displays y para establecer los límites de comparación.