

Instituto Tecnológico y de Estudios
Superiores de Occidente – ITESO



ITESO

Universidad Jesuita
de Guadalajara

Materia: Fundamentos de microprocesadores
y microcontroladores
Profesor: Álvaro Gutiérrez Arce

Práctica 2

Fecha: 11 de noviembre de 2021
Tema: Práctica 2: Alarma
Autores: Miriam Guadalupe Malta Reyes
Rodrigo Zamora Davalas

Lista de contenidos

Desarrollo técnico	2
Esquemático	4
Diagrama de flujo	5
Código del programa con comentarios	6
Miriam Guadalupe Malta Reyes	13
Rodrigo Zamora Dávalos	13
Referencias	14

Desarrollo técnico

Cuando dos o más dígitos se requieren para una aplicación, se debe usar multiplexión de dígitos. Dado que cada dígito requiere 7 pines de salida del microcontrolador, si se emplean dos o más dígitos, el número de pines restantes en el microcontrolador se ve limitado. Por lo tanto, se conectan todos los 7 segmentos de cada dígito en paralelo y el ánodo común de cada display se conecta a una salida del microcontrolador. Cuando se desea mostrar un número en un display, se activa la salida para ese display y se setean los bits necesarios para mostrar el número deseado. Después se apaga ese segmento, se actualiza la información para el siguiente display y se prende el siguiente display y así sucesivamente.

Las alarmas se han usado por siglos porque nos ayudan a resolver un problema: asegurarse de levantarse a tiempo. Su utilidad es indispensable y muchos de nosotros tendríamos problemas para llegar a tiempo al trabajo, escuela, entre otras responsabilidades sin un reloj con alarma. Por lo tanto, muchas personas dependen de un reloj con alarma para empezar el día. El objetivo de esta práctica es desarrollar un circuito capaz de sonar un pequeño buzzer cuando la hora del día coincida con la hora programada para la alarma.

El objetivo de nuestra práctica es diseñar un reloj con alarma que cuente con las siguientes especificaciones.

1. El reloj debe ser exacto; es decir, no se debe atrasar ni adelantar.
2. El sistema debe mostrar la hora (real o de la hora de alarma) en el formato HH:MM en displays de 7 segmentos.
3. El sistema debe poder ponerse a tiempo.
4. El sistema debe mostrar la hora 12:00:00 después de un reset. La hora de alarma inicializarla a las 00:00:00
5. El sistema debe contar con un interruptor para visualizar la hora real o la hora de alarma.
6. El sistema debe contar con un interruptor para hacer que la hora (real o de la alarma) se ponga a tiempo de manera ascendente a partir de los minutos.
7. El sistema debe mostrar siempre la hora (real o de la alarma), aun cuando se le esté actualizando.
8. El sistema debe sonar una chicharra por un minuto si se activa la alarma.
9. El sistema consta del siguiente HW:
 - 4 displays de 7 segmentos
 - 4 transistores y 4 resistencias para energizar cada uno de los displays
 - 7 resistencias para energizar cada uno de los segmentos
 - un interruptor para mostrar la hora real o la hora de la alarma
 - un push button para hacer que avance la cuenta del tiempo por minuto

- un push button para reiniciar la cuenta del tiempo
- un push button para reiniciar la cuenta de la alarma
- un buzzer para que suene la alarma
- circuito de reset
- circuito del cristal externo
- un led para señalar si lo que se ve es la hora real o la hora de alarma

10. El sistema consta del siguiente SW:

- un programa principal en el que se hacen las inicializaciones (registros y localidades de memoria), se verifica si lo que se va a desplegar es la hora real o la hora de alarma (pudiéndose utilizar en su lugar una interrupción) y esperar las interrupciones que hacen funcionar el dispositivo.
- una subrutina de interrupción para el tic de un minuto y actualización de la hora real en BCD y en 7 segmentos.
- una subrutina de interrupción para la multiplexación de los displays.
- una subrutina para poner a tiempo el reloj.

Para el armado de la alarma utilizamos:

- 1 - AT28C64B
- 2 - protoboards
- Alambre
- 5 - Pushbutton
- 1 - Resistencia de 8.2k ohms
- 1 - Capacitor de 10uF
- 1 - Cristal de Cuarzo de 12 MHz
- 2 - Capacitores de 33p
- 1 - LEDs
- 8 - Resistencias de 330 ohms
- 4 - Resistencias de 1k
- 4 - Transistores 2n2222a
- 7 - Resistencias de 10k
- 4 - Displays de 7 segmentos ánodo común
- 1 - Buzzer

Esquemático

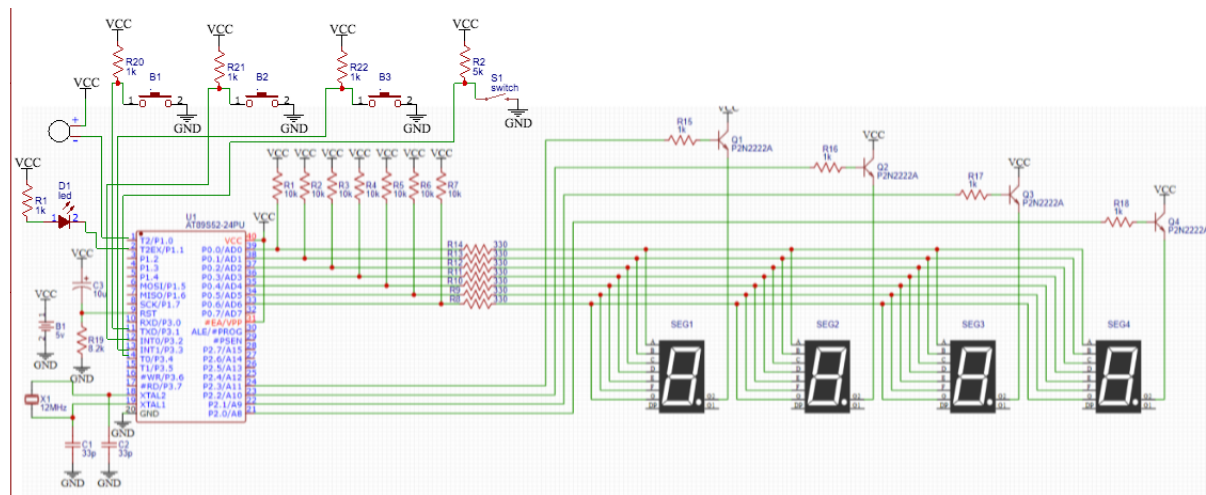
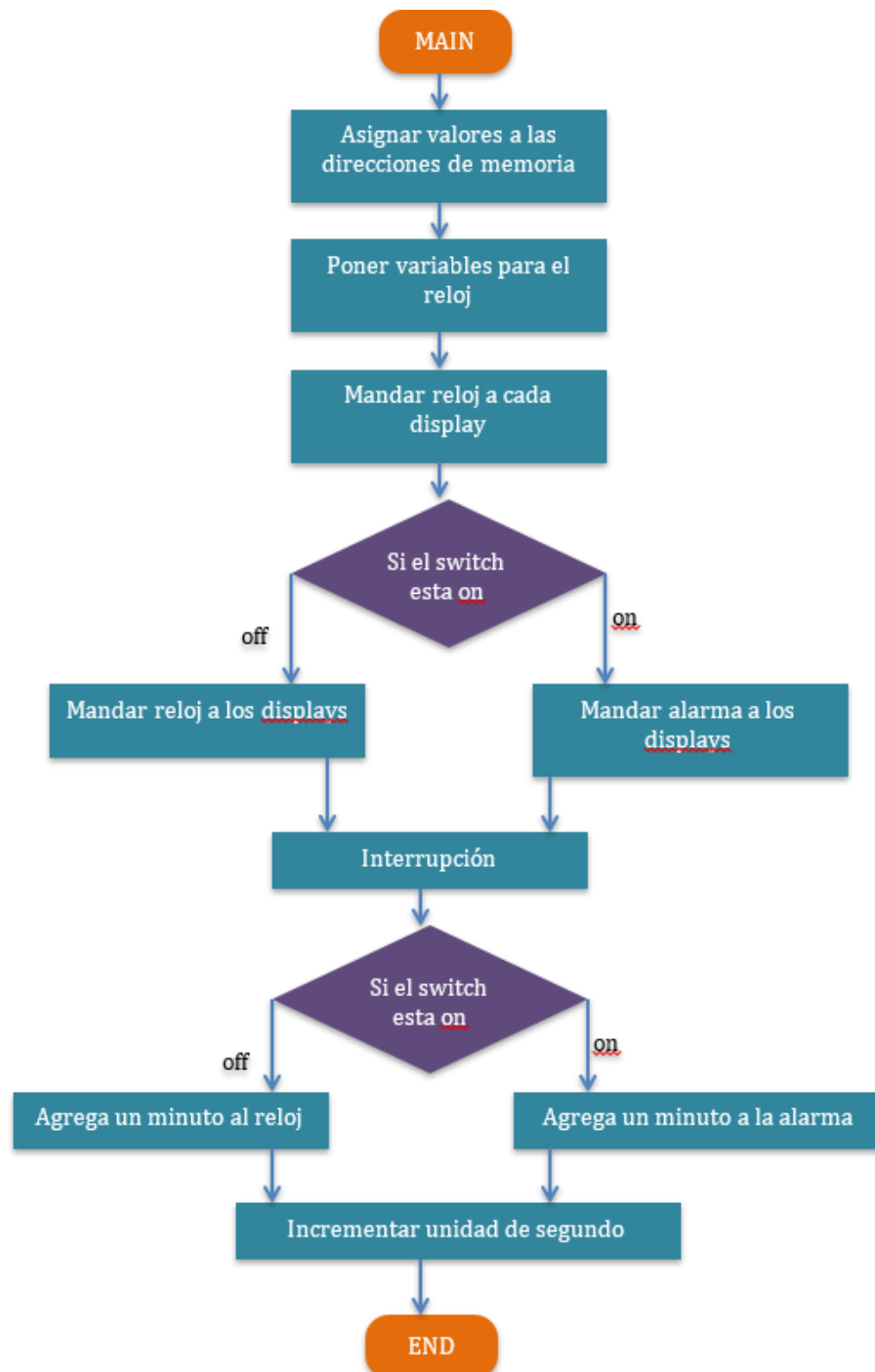


Diagrama de flujo



Código del programa con comentarios

```
ORG 0000H
JMP INICIO                                // Salto subrutina de inicio

ORG 000BH
JMP RTT0                                  // Interrupción timer 0

ORG 001BH
JMP RTT1                                  // Interrupción timer 1

ORG 0003H
JMP EXT0                                  // Interrupción externa 0

ORG 0013H
JMP EXT1                                  // Interrupción externa 1

ORG 0030H

INICIO:
MOV IP,#00001010B                        // Configuración prioridad de interrupciones
MOV IE,#10001111B                        // Activación de interrupciones

                                           // Reloj - Empezar el reloj en 12:00
MOV 32H,#0H                              // Unidad de segundo
MOV 33H,#0H                              // Decena de segundo
MOV 34H,#0H                              // Unidad de minuto
MOV 35H,#0H                              // Decena de minuto
MOV 36H,#2H                              // Unidad de hora
MOV 37H,#1H                              // Decena de hora

                                           // Alarma - Empezar la alarma en
00:00
MOV 52H,#0H                              // Unidad de minuto
MOV 53H,#0H                              // Decena de minuto
MOV 54H,#0H                              // Unidad de hora
MOV 55H,#0H                              // Decena de hora

MOV 48H,#0H                              // Debouncer

                                           // hgfedcba
                                           Decodificador 7 segmentos
MOV 3CH,#01000000b                      // 0
MOV 3DH,#01111001b                      // 1
MOV 3EH,#00100100b                      // 2
```

```

MOV 3FH,#00110000b           // 3
MOV 40H,#00011001b           // 4
MOV 41H,#00010010b           // 5
MOV 42H,#00000010b           // 6
MOV 43H,#01111000b           // 7
MOV 44H,#00000000b           // 8
MOV 45H,#00011000b           // 9

MOV P2, #11011111B           // Inicialización del puerto 0 - displays
MOV P0, #11111111B           // Inicialización del puerto 2 - transistores
MOV TMOD,#00010001B           // Configuración de timers
MOV TH0,#HIGH(-1926)          // Inicialización conteo inicial HIGH TIMER 0
MOV TL0,#LOW(-1926)           // Inicialización conteo inicial LOW TIMER 0
MOV TH1,#HIGH(-500)           // Inicialización conteo inicial HIGH TIMER 1
MOV TL1,#LOW(-500)            // Inicialización conteo inicial LOW TIMER 1
SETB TR0                      // Inicia conteo timer 0
MOV R2,#0H
MOV R6,#0H
SETB P1.0                     // Buzzer

ESPERA:
    JNB P3.1, CLEARALARM      // Botón para reiniciar la alarma
    JMP ALARM_1

CLEARALARM:
    MOV 52H,#0H                // Unidad de minuto
    MOV 53H,#0H                // Decena de minuto
    MOV 54H,#0H                // Unidad de hora
    MOV 55H,#0H                // Decena de hora
    ALARM_1:                   // Comparar unidad de minuto de alarma
        contra unidad de minuto de reloj, si es lo mismo, incrementar R6
        CLR A
        MOV A, 34H
        CJNE A, 52H, ALARM_2
        INC R6

    ALARM_2:                   // Comparar decena de minuto de alarma
        contra decena de minuto de reloj, si es lo mismo, incrementar R6
        CLR A
        MOV A, 35H
        CJNE A, 53H, ALARM_3
        INC R6

    ALARM_3:                   // Comparar unidad de hora de alarma
        contra decena de hora de reloj, si es lo mismo, incrementar R6
        CLR A

```



```

        MOV A, 36H
        CJNE A, 54H, ALARM_4
        INC R6

ALARM_4:                                     // Comparar decena de hora de alarma
contra decena de hora de reloj, si es lo mismo, incrementar R6
        CLR A
        MOV A, 37H
        CJNE A, 55H, ALARM_5
        INC R6

ALARM_5:                                     // Si R6 es igual a 4, significa que la hora de
alarma y la hora del reloj es la misma, por lo tanto hacemos sonar la alarma
        CLR A
        CJNE R6, #4, ALARM_SOUND
        MOV R6, #0
        CLR P1.0

ALARM_SOUND:
        MOV R6, #0
        SETB P1.0
        JMP ESPERA

RTT0:
        SETB TR1                                     // Subrutina de interrupción time 0 (Se
activa timer 1)
        CLR TR1                                     // Se desactiva timer 1
        MOV TH0, #HIGH(-1991)                       // Se actualizan valores high de cuenta timer 0
        MOV TL0, #LOW(-1991)                         // Se actualizan valores low de cuenta timer
0
        INC 48H
        MOV P2, #0H
        JNB P3.4, ALARMA                             // Leer el pin 3.4 para la alarma
        SETB P1.1                                     // Pin 1.1 para el reinicio del reloj

// Mandar a los displays el reloj
DISPLAY_1:
        MOV R1, #00000001B
        MOV A, 34H
        CJNE R2, #0H, DISPLAY_2
        JMP DISPLAYS

DISPLAY_2:
        MOV R1, #00000010B
        MOV A, 35H
        CJNE R2, #1H, DISPLAY_3
        JMP DISPLAYS

```

```

DISPLAY_3:
    MOV R1,#00000100B
    MOV A,36H
    CJNE R2,#2H,DISPLAY_4
    JMP DISPLAYS

DISPLAY_4:
    MOV R1,#00001000B
    MOV A,37H
    MOV R2,#0H
    CJNE R2,#3H,RESTART_DISPLAY
    JMP DISPLAYS

RESTART_DISPLAY:
    MOV R2,#0
    DEC R2

DISPLAYS:
    INC R2
    MOV P2,R1
    ADD A,#3CH
    MOV R0,A
    MOV P0,@R0
    RETI
// Salir de la subrutina

```

// Mandar a los displays la alarma

```

ALARMA:
    CLR P1.1
    AL_DISPLAY_1:
        MOV R1,#00000001B
        MOV A,52H
        CJNE R2,#0H,AL_DISPLAY_2
        JMP AL_DISPLAYS

    AL_DISPLAY_2:
        MOV R1,#00000010B
        MOV A,53H
        CJNE R2,#1H,AL_DISPLAY_3
        JMP AL_DISPLAYS

    AL_DISPLAY_3:
        MOV R1,#00000100B
        MOV A,54H
        CJNE R2,#2H,AL_DISPLAY_4
        JMP AL_DISPLAYS

    AL_DISPLAY_4:
        MOV R1,#00001000B
        MOV A,55H

```

```

        MOV R2,#0H
        CJNE R2,#3H,AL_RESTART_DISPLAY
        JMP AL_DISPLAYS
AL_RESTART_DISPLAY:
        MOV R2,#0
        DEC R2
AL_DISPLAYS:
        INC R2
        MOV P2,R1
        ADD A,#3CH
        MOV R0,A
        MOV P0,@R0
        RETI                                     // Salir de la subrutina

RTT1:
        MOV TH1,#HIGH(-500)                     // Subrutina
interrupción del timer1, 1s
        MOV TL1,#LOW(-500)
        UNISEG:
                INC 32H                           // Aumenta la
localidad de unidad de segundo
                MOV R3,32H                         // Mueve a R3 la
unidad de segundo
                CJNE R3,#0AH,ENDCLK
                MOV 32H,#0H                         // Mueve a unidad de
segundo el valor de 0H
        DECSEG:
                INC 33H                           // Aumenta la
localidad de decena de segundo
                MOV R3,33H                         // Mueve a R3 la
decena de segundo
                CJNE R3,#06H,ENDCLK
                MOV 33H,#0H                         // Mueve a decena de
segundo el valor de 0H
        UNIMIN:
                INC 34H                           // Aumenta la
localidad de unidad de minuto
                MOV R3,34H                         // Mueve a R3 la
unidad de minuto
                CJNE R3,#0AH,ENDCLK
                MOV 34H,#0H                         // Mueve a unidad de
minuto el valor de 0H
        DECMIN:

```

	INC 35H	// Aumenta la
localidad de decena de minuto	MOV R3,35H	// Mueve a R3 la
decena de minuto	CJNE R3,#06H,ENDCLK	
	MOV 35H,#0H	// Mueve a decena de
minuto el valor de 0H	UNIHOUR:	
	INC 36H	// Aumenta la
localidad de unidad de hora	MOV R3,36H	// Mueve a R3 la
unidad de hora	CJNE R3,#0AH,ENDCLK	
	MOV 36H,#0H	// Mueve a unidad de
hora el valor de 0H	DECHORA:	
	INC 37H	// Aumenta la
localidad de decena de hora	MOV R3,37H	// Mueve a R3 la
decena de hora	CJNE R3,#03H,ENDCLK	
	MOV 37H,#0H	// Mueve a decena de
hora el valor de 0H	ENDCLK:	
	RETI	// Salir de la subrutina
EXT0:	MOV 48H,#0H	
ADMIN:		// Agregar
minuto	MOV R4,48H	// Debouncer para
tener un tiempo de espera entre cada agregar minuto	CJNE R4,#50H,ADMIN	
	JNB P3.4, ADDMINALARMA	// Si está el switch en alarma,
agregar minuto a la alarma	JMP UNIMIN	
	RETI	
ADDMINALARMA:		
ADMIN_UNIMIN:	INC 52H	// Aumenta la
localidad de unidad de minuto		

	MOV R3,52H	// Mueve a R3 la
unidad de minuto	CJNE R3,#0AH,ADDMIN_ENDCLK	
	MOV 52H,#0H	// Mueve a unidad de
minuto el valor 0H	ADDMIN_DECMIN:	
	INC 53H	// Aumenta la
localidad de decena de minuto	MOV R3,53H	// Mueve a R3 la
decena de minuto	CJNE R3,#06H,ADDMIN_ENDCLK	
	MOV 53H,#0H	// Mueve a decena de
minuto el valor 0H	ADDMIN_UNIHOURL:	
	INC 54H	// Aumenta la
localidad de unidad de hora	MOV R3,54H	// Mueve a R3 la
unidad de hora	CJNE R3,#0AH,ADDMIN_ENDCLK	
	MOV 54H,#0H	// Mueve a unidad de
hora el valor de 0H	ADDMIN_DECHORA:	
	INC 55H	// Aumenta la
localidad de decena de hora	MOV R3,55H	// Mueve a R3 la
decena de hora	CJNE R3,#03H,ADDMIN_ENDCLK	
	MOV 55H,#0H	// Mueve a decena de
hora el valor de 0H	ADDMIN_ENDCLK:	
	RETI	
EXT1:		
	MOV 32H,#0H	// Unidad de segundo
	MOV 33H,#0H	// Decena de segundo
	MOV 34H,#0H	// Unidad de minuto
	MOV 35H,#0H	// Decena de minuto
	MOV 36H,#2H	// Unidad de hora
	MOV 37H,#1H	// Decena de hora
	RETI	
	END	

Conclusiones individuales

Miriam Guadalupe Malta Reyes

Con esta práctica aprendí y comprendí bastantes cosas. Me ayudó a seguir practicando cómo usar ensamblador. Aprender desde cómo usar un arduino para programar el microcontrolador, como hacer que se vieran algo en los displays, como hacer un multiplexeado de dígitos, como hacer una alarma, como hacer que se incrementara con botones, como poner un switch, como hacer interrupciones, que es un debouncer, como poner un buzzer.

Rodrigo Zamora Dávalos

Esta práctica me ayudó mucho a entender mejor como programar en ensamblador. Tenía muchas dudas pero ya con esta práctica las pude resolver, también aprendí a usar las interrupciones para los botones, y también que teníamos que usar un debouncer para que al momento de presionar el botón de agregar minuto, no se fuera tan rápido.

Referencias

Manish K Patel (2014). The 8051 Microcontroller Based Embedded Systems. McGraw Hill Education