Universidad del País Vasco
Euskal Herriko Unibertsitatea
FACULTY OF ENGINEERING BILBAO
UNIVERSITY OF THE BASQUE COUNTRY

Arquitectura de Computadores:

Práctica Final de Laboratorio 2022-2023

Departamento:
Tecnología Electrónica

Titulación:
Grado en Informática de Gestión y Sistemas de Información

Xabier Gabiña Barañano
Ainhize Martínez Duran

Contenido

INTRODUCCION

En esta práctica se pretende realizar el software de control de una **placa de inducción de un fogón** mediante la utilización del **microcontrolador 80C552 de Philips**.

Nuestra tarea será programar diversas funciones tales como dos pulsadores capacitivos, un **DISPLAY**, un **led**, un **zumbador**, un **ADC**, un **PWM** y un **TIMER** con los conocimientos obtenidos en las clases de practica de aula y de teoría.

Todo esto como he dicho antes se hará haciendo uso del **microcontrolador 80C552**, un derivado del **80C51**. Y será programado y testado en el entorno de desarrollo "**Keil uVision 2**".

TABLAS Y DIAGRAMAS

Todos estos diagramas estarán incluidos en la carpeta de la entrega para mayor calidad del visionado.

TABLAS DE ESTADO/EVENTO/ACCION

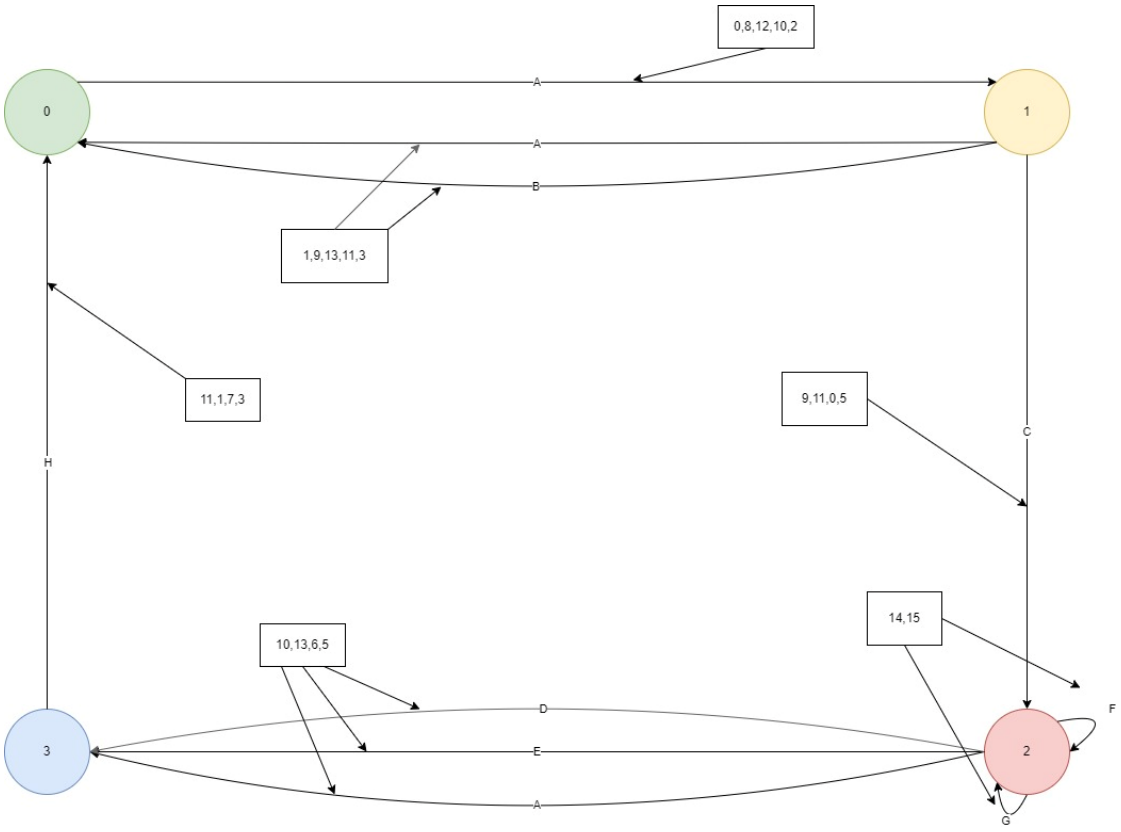
TABLA DE ESTADOS	
COD	*NOMBRE*
0	REPOSO
1	ESPERA
2	CALENTAR
3	TRANSICION

TABLA DE EVENTOS	
COD	*NOMBRE*
A	SE HA PULSADO ON/OFF
B	HAN PASADO 15s SIN RECIPIENTE
C	SE HA COLOCADO RECIPIENTE
D	HAN PASADO 60s SIN AUMENTAR POTENCIA
E	HAN PASADO 30s SIN RECIPIENTE
F	SE HA PULSADO EL BOTON +
G	SE HA PULSADO EL BOTON -
H	LA TEMPERATURA ES INFERIOR A 40

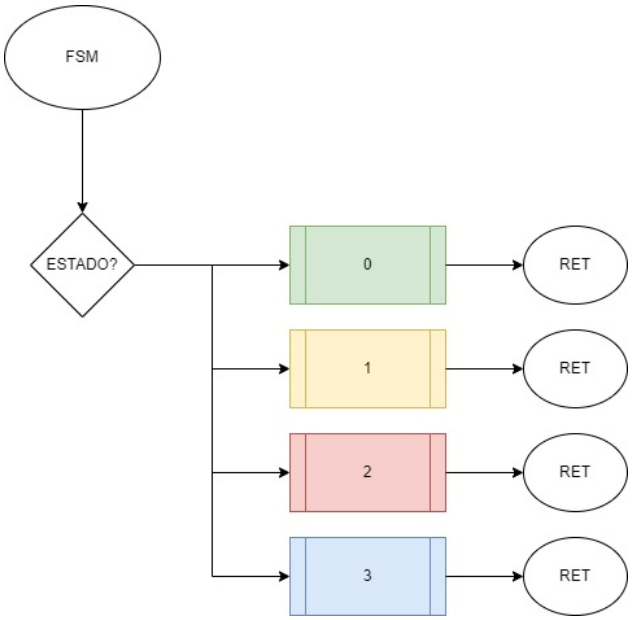
TABLA DE EVENTOS POR ESTADO		
COD ESTADO	*COD EVENTO* EN ESTADO	*COD EVENTO*
0	1	A
1	1	A, B
	2	C
2	1	A, D, E
	2	G
	3	F
3	1	H

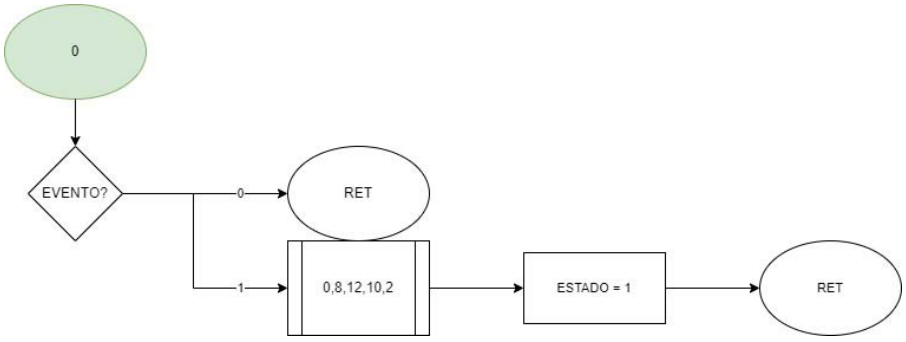
TABLA DE ACCIONES		
COD	*NOMBRE*	*NOMBRE EN EL CODIGO*
0	ENCENDER DISPLAY	ENCENDER_DISPLAY
1	APAGAR DISPLAY	APAGAR_DISPLAY
2	ENCENDER TIMER	ENCENDER_TIMER
3	APAGAR TIMER	APAGAR_TIMER
4	ENCENDER PWM	ENCENDER_PWM
5	APAGAR PWM	APAGAR_PWM
6	ENCENDER ADC	ENCENDER_ADC
7	APAGAR ADC	APAGAR_ADC
8	ENCENDER PARPADEO	ENCENDER_PARPADEO
9	APAGAR PARPADEO	APAGAR_PARPADEO
10	ENCENDER ZUMBADOR	ENCENDER_ZUMBADOR
11	APAGAR ZUMBADOR	APAGAR_ZUMBADOR
12	ENCENDER LED	ENCENDER_LED
13	APAGAR LED	APAGAR_LED
14	+ VALOR FOGÓN	INCREMENTAR_FOGON
15	- VALOR FOGÓN	DECREMENTAR_FOGON
16	ACTUALIZAR DISPLAY	UPDATE_DISPLAY

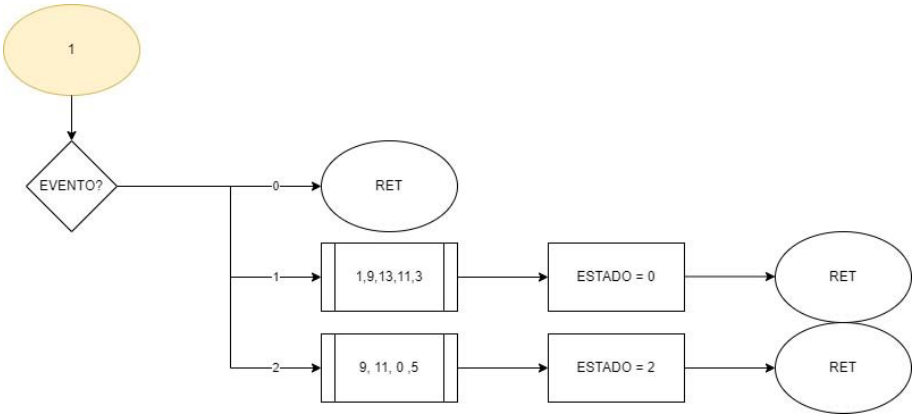
DIAGRAMA DE ESTADO/EVENTO/ACCION

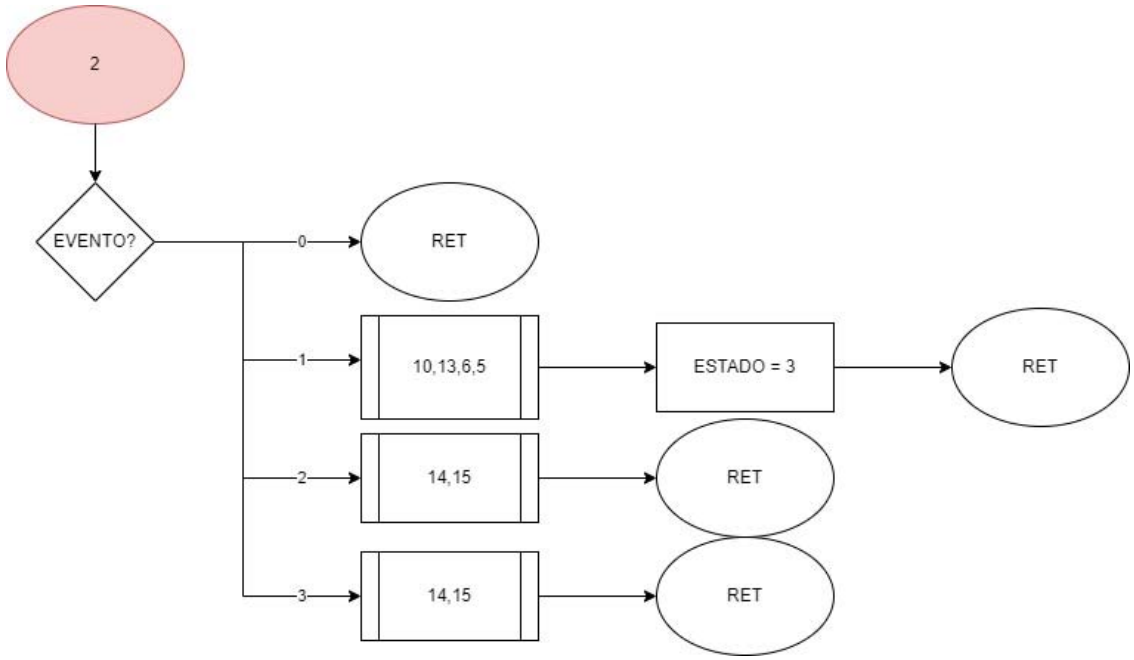


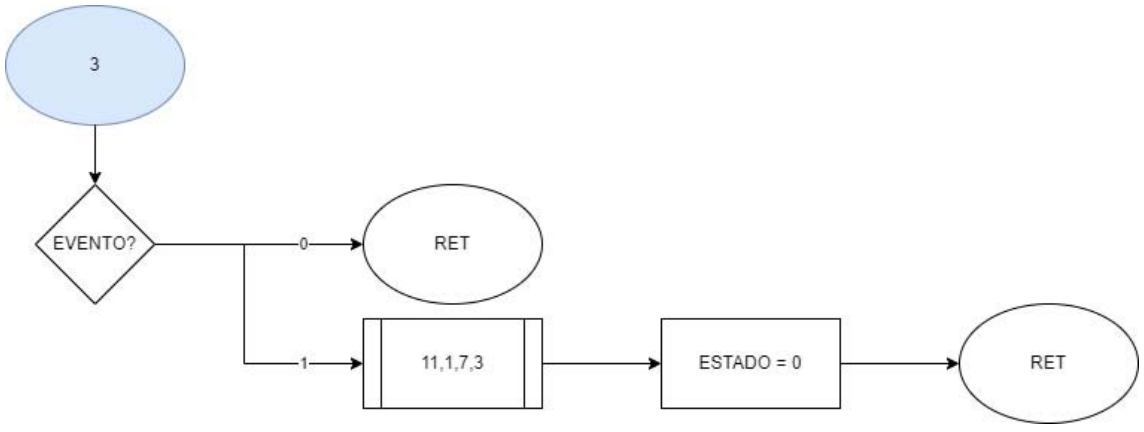
DIAGRAMAS DE FLUJO

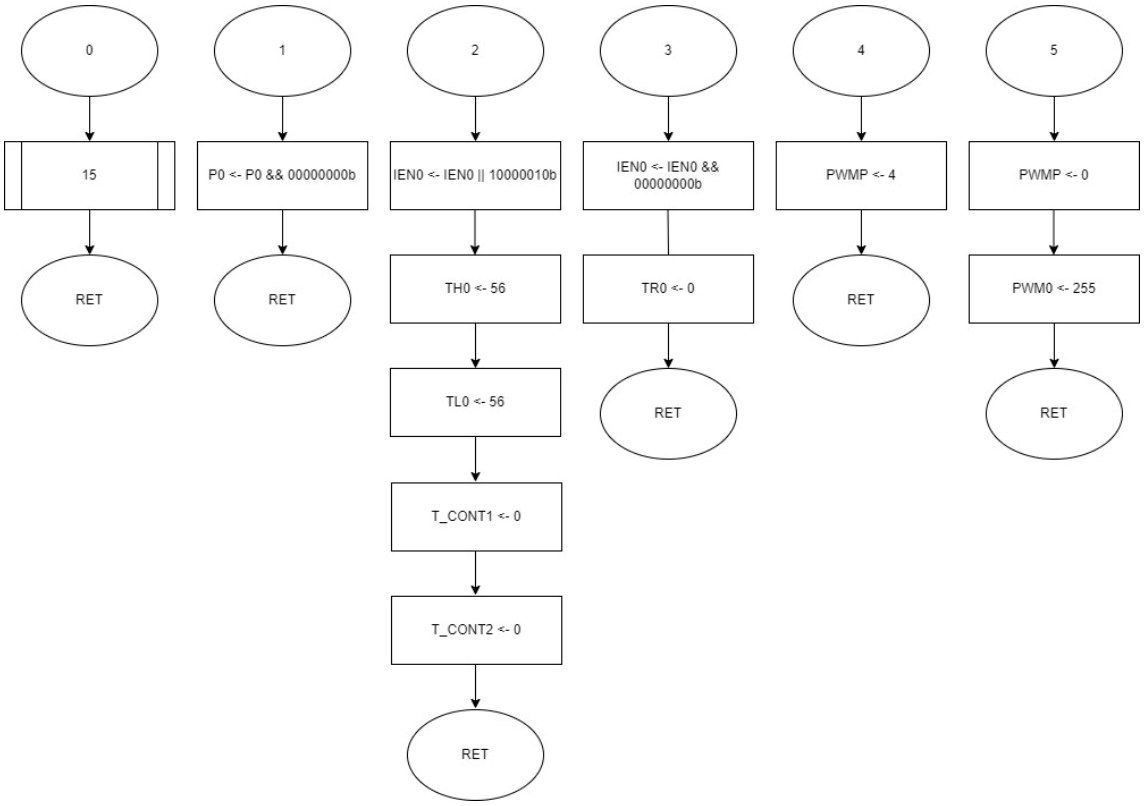


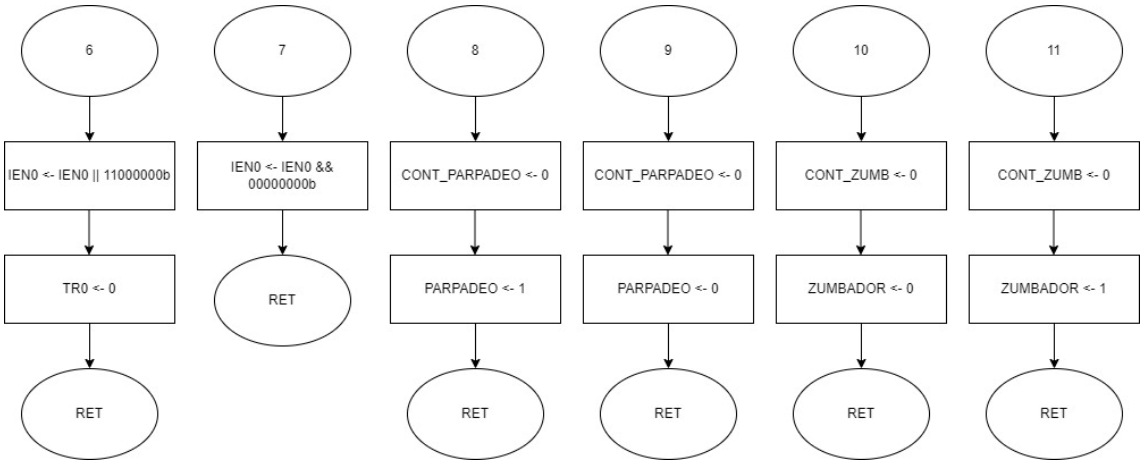


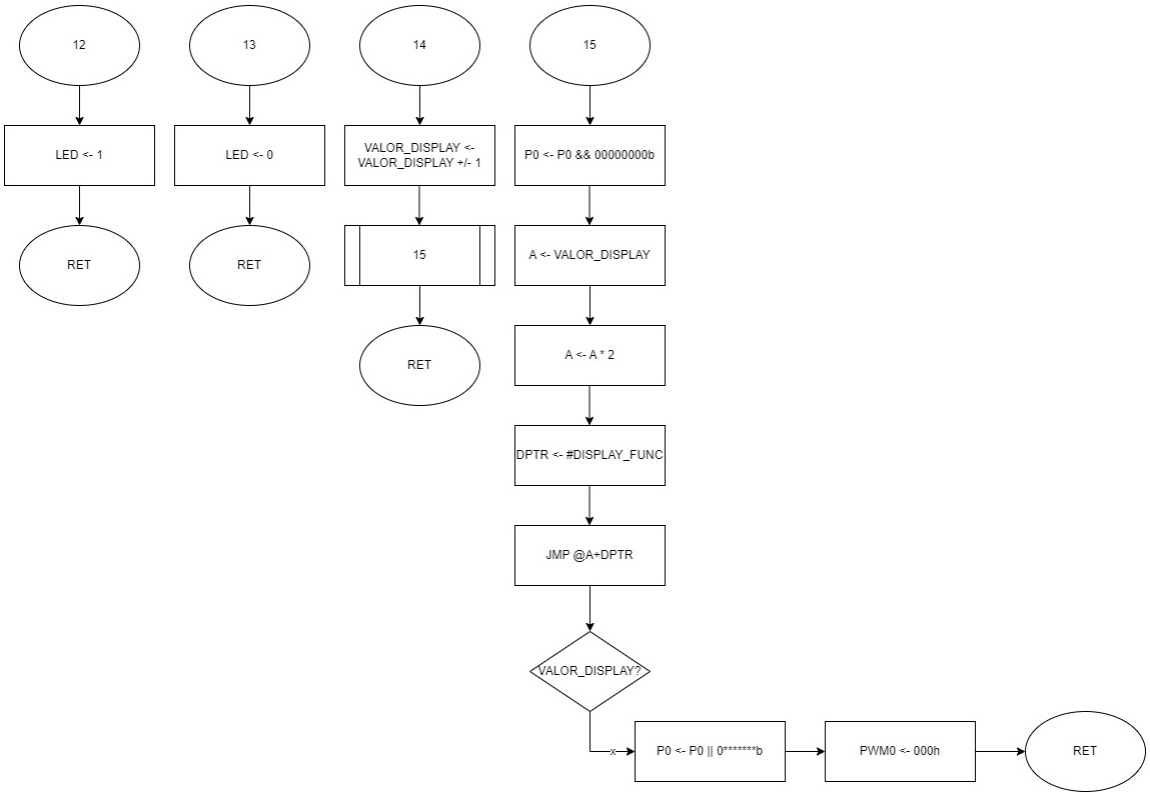



2
EVENTO?
0
RET
1
10,13,6,5
ESTADO = 3
RET
2
14,15
RET
3
14,15
RET


3
EVENTO?
0
RET
1
11,1,7,3
ESTADO = 0
RET


0
15
RET
1
P0 <- P0 && 00000000b
RET
2
IEN0 <- IEN0 || 10000010b
TH0 <- 56
TL0 <- 56
T_CONT1 <- 0
T_CONT2 <- 0
RET
3
IEN0 <- IEN0 && 00000000b
TR0 <- 0
RET
4
PWMP <- 4
RET
5
PWMP <- 0
PWM0 <- 255
RET


6
IEN0 <- IEN0 || 11000000b
TR0 <- 0
RET
7
IEN0 <- IEN0 && 00000000b
RET
8
CONT_PARPADEO <- 0
PARPADEO <- 1
RET
9
CONT_PARPADEO <- 0
PARPADEO <- 0
RET
10
CONT_ZUMB <- 0
ZUMBADOR <- 0
RET
11
CONT_ZUMB <- 0
ZUMBADOR <- 1
RET


12
13
14
15
LED <- 1
LED <- 0
VALOR_DISPLAY <- VALOR_DISPLAY +/- 1
P0 <- P0 && 00000000b
RET
RET
15
A <- VALOR_DISPLAY
RET
A <- A * 2
DPTR <- #DISPLAY_FUNC
JMP @A+DPTR
VALOR_DISPLAY?
x
P0 <- P0 || 0*******b
PWM0 <- 000h
RET

CALCULOS Y COMENTARIOS

ADC

La temperatura del fogón de inducción se mide mediante el canal 0 del convertidor analógico-digital del **80C552**. El valor que este nos arroja es proporcional a la temperatura que el sensor lee.

Para este trabajo necesitamos conocer el valor que arrojara cuando la temperatura del fogón sea de **40** y de **80** grados centígrados. Sabemos que el sensor sigue una progresión de 10mV por cada Cº luego podemos calcular todo lo necesario.

$10\frac{mV}{Cº} * 40Cº \rightarrow x = 0.4V$ $10\frac{mV}{Cº} * 80Cº \rightarrow x = 0.8V$

Para calcular el resultado del **SFR ADCH** podemos usar la siguiente formula.

$$256 \cdot \frac{Vin - AVref}{AVref - AVref}$$

En la cual tras sustituir los valores obtendremos los resultados.

$256 \cdot \frac{0.4-0}{5-0} = 20.48 \sim \mathbf{20}\boldsymbol{d}$ $256 \cdot \frac{0.8-0}{5-0} = 40.96 \sim \mathbf{40}\boldsymbol{d}$

TIMER

En el **80C552** tenemos varios "**Timers**" a nuestra disposición con diferentes modos y funciones especiales. Para nuestro proyecto hemos usado exclusivamente el **Timer1** en modo de **8bits con auto recarga**.

Como se nos especifica en la documentación del programa contamos con un cristal de cuarzo que oscila a **24MHz**. Sabiendo que por cada **doce ciclos de reloj** tenemos **un ciclo maquina** podemos calcular que la frecuencia de instrucción es de **2MHz**, es decir, se tarda **0,5µs** en ejecutar cada instrucción. Con ello queremos que el "**timer**" active una "**flag**" que nos indique el paso de 100ms para lo cual realizamos los siguientes cálculos:

To = 0,5µs = 0,0005ms
Te = 100ms
Te/To = 200000
200000 = 200 * 125 * 8
200 -> 256-200 = **56** -> Precarga de timer (**TH0** y **TL0**)
125 -> Contador 1 del programa
8 -> Contador 2 del programa

PWM

La temperatura del fogón de inducción viene controlada por un **PWM.** Dependiendo el nivel en el que se encuentre el fogón dará un porcentaje de potencia equivalente, es decir, en el nivel 0 dará 0% de potencia, en el nivel 1 dará 10% de potencia y así hasta la P que será el 100% de potencia. Por lo tanto, podemos calcular los valores que tendrá **PWM0.**

$$D = \frac{255 - PWM0}{255} \rightarrow PWM0 = 255(1 - D)$$

$0\% \rightarrow 255(1-0) = \mathbf{255}$
$10\% \rightarrow 255(1-0.1) = 229.5 \sim \mathbf{229}$
$20\% \rightarrow 255(1-0.2) = \mathbf{204}$
$30\% \rightarrow 255(1-0.3) = 178.5 \sim \mathbf{178}$
$40\% \rightarrow 255(1-0.4) = \mathbf{153}$
$50\% \rightarrow 255(1-0.5) = 127.5 \sim \mathbf{127}$
$60\% \rightarrow 255(1-0.6) = \mathbf{102}$
$70\% \rightarrow 255(1-0.7) = 76.5 \sim \mathbf{76}$
$80\% \rightarrow 255(1-0.8) = \mathbf{51}$
$90\% \rightarrow 255(1-0.9) = 25.5 \sim \mathbf{25}$
$100\% \rightarrow 255(1-1) = \mathbf{0}$

Ahora nos indican que el **PWM** tiene una frecuencia de conmutación de 10kHz por lo que podemos calcular el valor del **PWMP** usando la siguiente formula:

$$Fpwm = \frac{Fosc}{2 \cdot (1 + PWMP) \cdot 255}$$

Sabiendo que la **Fosc** es la del reloj, es decir, 24Mhz y **Fpwm** son los 10kHz que se piden sustituimos:

$$10kHz = \frac{24Mhz}{2 \cdot (1 + PWMP) \cdot 255} = 3{,}70588 \sim 4$$

COMENTARIOS

En general el proyecto nos ha parecido que tenía una dificultad bastante aceptable y asequible. En nuestro caso íbamos programando cada cosa a medida que dábamos su teoría en clase por lo que no nos ha resultado demasiado difícil ningún apartado quitando el hecho de resolver los cálculos de los diferentes módulos como el **ADC**, **TIMER** y **PWM**.

Si tuviese que resaltar algo me centraría sobre todo el en **TIMER** el cual se usa mucho en todas partes del programa y tuvimos alguna dificultad para elegir la frecuencia del reloj, pero nada imposible de hacer.

CODIGO

```
;=================================================================================================;
                                ;VARIABLES GLOBALES
ESTADO EQU 0x20                 ;BYTE DONDE SE GUARDA EL ESTADO
EVENTO EQU 0x21                 ;BYTE DONDE SE GUARDA EL EVENTO
VALOR_DISPLAY EQU 0x22          ;BYTE DEL VALOR QUE SE MUESTRA EN EL DISPLAY
T_CONT1 EQU 0x23                ;BYTE DONDE SE GUARDA EL CONTADOR[1] USADO EN EL TIMER
T_CONT2 EQU 0x24                ;BYTE DONDE SE GUARDA EL CONTADOR[2] USADO EN EL TIMER
TICK EQU 0x25.0                 ;BIT DONDE SE GUARDA LA FLAG DE AVISO PARA 100MS
PARPADEO EQU 0x25.1             ;BIT DONDE SE GUARDA LA FLAG DE AVISO DEL ESTADO DE PARPADEO
FADC EQU 0X25.2                 ;BIT DONDE SE GUARDA LA FLAG DE AVISO DEL ADC
CONT_ZUMB EQU 0x26              ;CONTADOR USADO EN EL ZUMBADOR
CONT_PARPADEO EQU 0x27          ;CONTADOR USADO EN EL PARPADEO

ADCON EQU 0xC5                  ;BYTE DE CONFIGURACION DE ADC
ADCH EQU 0xC6                   ;BYTE DONDE SE GUARDA EL VALOR DE LECTURA DEL ADC
PWMP EQU 0xFE                   ;BYTE DEL PRESCALER DEL PWM
PWM0 EQU 0xFC                   ;BYTE DONDE SE GUARDA EL DUTY CICLE
IEN0 EQU 0xA8                   ;BYTE DE CONFIGURACION DE LAS INTERRUPCIONES

                                ;PUERTOS
LED EQU P2.0                    ;PUERTO AL QUE ESTA CONECTADO EL LED
ZUMBADOR EQU P2.1               ;PUERTO AL QUE ESTA CONECTADO EL ZUMBADOR
PLUS EQU P3.0                   ;PUERTO AL QUE ESTA CONECTADO EL PULSADOR +
MINUS EQU P3.1                  ;PUERTO AL QUE ESTA CONECTADO EL PULSADOR -
SWITCH EQU P3.2                 ;PUERTO AL QUE ESTA CONECTADO EL PULSADOR ON/OFF
SENSOR EQU P3.5                 ;PUERTO AL QUE ESTA CONECTADO EL SENSOR

                                ;VARIABLES DE ESTADOS
ES1_CONT_MS EQU 0x2A            ;CONTADOR USADO EN EL ESTADO1 PARA CONTAR LOS TICKS PASADOS
ES1_CONT_S EQU 0x2B             ;CONTADOR USADO EN EL ESTADO1 PARA CONTAR LOS S PASADOS

ES2_CONT_MS EQU 0x2C            ;CONTADOR USADO EN EL ESTADO2 PARA CONTAR LOS TICKS PASADOS
ES2_CONT1_S EQU 0x2D            ;CONTADOR USADO EN EL ESTADO2 PARA CONTAR LOS S PASADOS
                                ;SIN SENSOR!!!
ES2_CONT2_S EQU 0x2E            ;CONTADOR USADO EN EL ESTADO2 PARA CONTAR LOS S PASADOS
                                ;CON SENSOR!!!

;AJUSTES TIMER
TIMER_CONT_1 EQU 125d           ;DEFAULT = 125
TIMER_CONT_2 EQU 8d             ;DEFAULT = 8
                                ;NORMALMENTE EL TIMER ACTIVA LA FLAG "TICK" CADA 100MS
                                ;PARA OBTENER LA FLAG DIEZ VECES MAS RAPIDO ABRIA QUE
                                ;USAR 50 y 2 RESPECTIVAMENTE AUNQUE EXISTEN
                                ;MAS COMBINACIONES COMO 100 Y 1

;=========================================INICIO=========================================;
ORG 0X0000
        AJMP INICIO

ORG 0x007b
INICIO:
        ACALL INIT
MAIN:
        ACALL FSM
        AJMP MAIN
FSM:
        MOV A, ESTADO
        RL A
        MOV DPTR, #EST_TAB
        JMP @A+DPTR
EST_TAB:
        AJMP ESTADO0            ;REPOSO
        AJMP ESTADO1            ;ESPERA
        AJMP ESTADO2            ;CALENTAR
        AJMP ESTADO3            ;TRANSICION
;=========================================INIT===========================================;
INIT:
                                ;VALORES GLOBALES
        MOV ESTADO, #0
        MOV EVENTO, #0
        MOV VALOR_DISPLAY, #0
        MOV T_CONT1, #0
```

```
        MOV T_CONT2, #0
        MOV CONT_ZUMB, #0
        MOV CONT_PARPADEO, #0
        CLR PARPADEO

                                ;PUERTOS ENTRADA
        SETB PLUS
        SETB MINUS
        SETB SWITCH
        SETB SENSOR

                                ;PUERTOS SALIDA
        CLR P0.0
        CLR P0.1
        CLR P0.2
        CLR P0.3
        CLR P0.4
        CLR P0.5
        CLR P0.6
        CLR P0.7
        CLR LED
        CLR ZUMBADOR
        SETB ZUMBADOR           ;ACTIVO A NIVEL BAJO!

                                ;TIMER EN MODO 8BIT AUTORECARGA
        ORL TMOD, #00000010b
        ANL TMOD, #11110010b
        CLR TICK

                                ;VARIABLES DE ESTADOS
        MOV ES1_CONT_MS, #0
        MOV ES1_CONT_S, #0

        MOV ES2_CONT_MS, #0
        MOV ES2_CONT1_S, #0
        MOV ES2_CONT2_S, #0

        RET
;=======================================ESTADO0_REPOSO=======================================;
ESTADO0:
        ACALL ES0_GEN_EV
        MOV A, EVENTO
        RL A
        MOV DPTR, #ES0_EV_TAB
        JMP @A+DPTR
ES0_EV_TAB:
        AJMP ES0_EV0
        AJMP ES0_EV1
        AJMP ES0_EV2
ES0_EV0:                        ;EVENTO VACIO
        RET
ES0_EV1:                        ;EVENTO IDL
        AJMP ACTIVAR_IDL
        RET
ES0_EV2:                        ;EVENTO ESPERA
        MOV VALOR_DISPLAY, #0

        ACALL UPDATE_DISPLAY
        ACALL ENCENDER_PARPADEO
        ACALL ENCENDER_LED
        ACALL ENCENDER_ZUMBADOR
        ACALL ENCENDER_TIMER

        MOV ESTADO, #1
        RET
ES0_GEN_EV:
        MOV C, SWITCH
        JNC ES0_SWITCH          ;COMPROBAR SI SE HA PULSADO EL BOTON
        MOV EVENTO, #1          ;SIEMPRE SE DA EL EVENTO 0 SI NO SE HA PULSADO EL SWITCH
        RET
ES0_SWITCH:                     ;SI EL SWITCH SE HA PULSADO PASA SE DA EL EVENTO 1
        MOV EVENTO, #2
        RET
;=======================================ESTADO1_ESPERA=======================================;
```

```
ESTADO1:
        ACALL ES1_GEN_EV
        MOV A, EVENTO
        RL A
        MOV DPTR, #ES1_EV_TAB
        JMP @A+DPTR
ES1_EV_TAB:
        AJMP ES1_EV0
        AJMP ES1_EV1
        AJMP ES1_EV2
ES1_EV0:                        ;EVENTO VACIO
        RET
ES1_EV1:                        ;EVENTO REPOSO
                                ;RESETEAR VARIABLES DE ESTADO
        MOV ES1_CONT_MS, #0
        MOV ES1_CONT_S, #0
        MOV CONT_ZUMB, #0
        MOV CONT_PARPADEO, #0

        ACALL APAGAR_DISPLAY
        ACALL APAGAR_PARPADEO
        ACALL APAGAR_ZUMBADOR
        ACALL APAGAR_LED
        ACALL APAGAR_TIMER

        MOV ESTADO, #0
        RET
ES1_EV2:                        ;EVENTO CALENTAR
                                ;RESETEAR VARIABLES DE ESTADO
        MOV ES1_CONT_MS, #0
        MOV ES1_CONT_S, #0
        MOV CONT_ZUMB, #0
        MOV CONT_PARPADEO, #0

        ACALL APAGAR_PARPADEO
        ACALL ENCENDER_DISPLAY
        ACALL APAGAR_ZUMBADOR
        ACALL ENCENDER_PWM

        MOV ESTADO, #2
        RET
ES1_GEN_EV:

        MOV C, SWITCH
        JNC ES1_SWITCH

        MOV C, SENSOR
        JC ES1_SENSOR

        MOV C, TICK
        JC ES1_TICK
        MOV EVENTO, #0
        RET
ES1_SWITCH:                     ;SE DETECTA ON/OFF
        MOV EVENTO, #1
        RET
ES1_SENSOR:                     ;SE DETECTA RECIPIENTE
        MOV EVENTO, #2
        RET
ES1_TICK:                       ;PASA 100MS SIN RECIPIENTE
        CLR TICK
        INC ES1_CONT_MS
        MOV A, #10
        CLR C
        SUBB A, ES1_CONT_MS
        JZ ES1_1S
        RET
ES1_1S:                         ;PASA 1S SIN RECIPIENTE
        MOV ES1_CONT_MS, #0
        INC ES1_CONT_S
        MOV A, #15
        CLR C
        SUBB A, ES1_CONT_S
        JZ ES1_15S
```

```
        RET
ES1_15S:                        ;PASA 15S SIN RECIPIENTE
        MOV ES1_CONT_S, #0
        MOV EVENTO, #1
        RET
;====================================ESTADO2_CALENTAR====================================;
ESTADO2:
        ACALL ES2_GEN_EV
        MOV A, EVENTO
        RL A
        MOV DPTR, #ES2_EV_TAB
        JMP @A+DPTR
ES2_EV_TAB:
        AJMP ES2_EV0
        AJMP ES2_EV1
        AJMP ES2_EV2
        AJMP ES2_EV3
ES2_EV0:                        ;EVENTO VACIO
        RET
ES2_EV1:                        ;EVENTO TRANSICION
        MOV ES2_CONT_MS, #0
        MOV ES2_CONT1_S, #0
        MOV ES2_CONT2_S, #0

        ACALL ENCENDER_ZUMBADOR
        ACALL APAGAR_LED
        ACALL ENCENDER_ADC
        ACALL APAGAR_PWM

        MOV ESTADO, #3
        RET
ES2_EV2:                        ;EVENTO PULSAR BOTON MENOS
        MOV A, #0
        CLR C
        SUBB A, VALOR_DISPLAY
        JZ ES2_NOTHING

        ACALL DECREMENTAR_FOGON

        RET
ES2_EV3:                        ;EVENTO PULSAR BOTON MAS
        MOV A, #10
        CLR C
        SUBB A, VALOR_DISPLAY
        JZ ES2_NOTHING

        ACALL INCREMENTAR_FOGON

        RET
ES2_GEN_EV:

        MOV C, SWITCH
        JNC ES2_SWITCH          ;SALTO SI SE PULSA EL BOTON ON/OFF

        MOV C, MINUS            ;SALTO SI SE PULSA EL BOTON -
        JC ES2_MINUS

        MOV C, PLUS             ;SALTO SI SE PULSA EL BOTON +
        JC ES2_PLUS

        MOV C, TICK             ;SALTO SI HAN PASADO 100MS
        JC ES2_TICK

        MOV EVENTO, #0
        RET
ES2_SWITCH:                     ;SE HA PULSADO EL BOTON ON/OFF
        MOV EVENTO, #1
        RET
ES2_MINUS:                      ;SE HA PULSADO EL BOTON DE MENOS
        MOV EVENTO, #2
        RET
ES2_PLUS:                       ;SE HA PULSADO EL BOTON DE MAS
        MOV EVENTO, #3
        RET
```

```
ES2_TICK:                       ;PASA 100MS
        CLR TICK
        INC ES2_CONT_MS
        MOV A, #10
        CLR C
        SUBB A, ES2_CONT_MS
        JZ ES2_1S
        RET
ES2_1S:                         ;PASA 1S
        MOV ES2_CONT_MS, #0
        ACALL ES2_SENSOR
        RET
ES2_30S:                        ;PASAN 30S
        MOV ES2_CONT1_S, #0
        MOV ES2_CONT2_S, #0
        MOV EVENTO, #1
        RET
ES2_60S:                        ;PASAN 60S
        MOV ES2_CONT1_S, #0
        MOV ES2_CONT2_S, #0
        MOV EVENTO, #1
ES2_SENSOR:                     ;DICE SI EL RECIPIENTE ESTA O NO
        MOV C, SENSOR
        JC ES2_SENSOR_ON
        JNC ES2_SENSOR_OFF
ES2_SENSOR_ON:                  ;SI ESTA RECIPIENTE SE CUENTA HASTA 60S
        MOV A, VALOR_DISPLAY
        JNZ ES2_NOTHING
        MOV ES2_CONT1_S, #0
        INC ES2_CONT2_S
        MOV A, #60
        CLR C
        SUBB A, ES2_CONT2_S
        JZ ES2_60S
        RET
ES2_SENSOR_OFF:                 ;NO ESTA RECIPIENTE SE CUENTA HASTA 30S
        MOV ES2_CONT2_S, #0
        INC ES2_CONT1_S
        MOV A, #30
        CLR C
        SUBB A, ES2_CONT1_S
        JZ ES2_30S
        RET
ES2_NOTHING:                    ;RESET DE LOS CONTADORES
        MOV ES2_CONT1_S, #0
        MOV ES2_CONT1_S, #0
        RET
;====================================ESTADO3_TRANSICION====================================;
ESTADO3:
        ACALL ES3_GEN_EV
        MOV A, EVENTO
        RL A
        MOV DPTR, #ES3_EV_TAB
        JMP @A+DPTR
ES3_EV_TAB:
        AJMP ES3_EV0
        AJMP ES3_EV1
ES3_EV0:                        ;EVENTO VACIO
        RET
ES3_EV1:                        ;EVENTO ESPERA
        ACALL APAGAR_TIMER
        ACALL APAGAR_ZUMBADOR
        ACALL APAGAR_DISPLAY
        ACALL APAGAR_ADC

        MOV ESTADO, #0
        RET
ES3_GEN_EV:
        MOV C, FADC
        JC ES3_ADC              ;EL ADC HA TERMINADO LA CONVERSION
        MOV EVENTO, #0
        RET
ES3_ADC:                        ;UNA VEZ TERMINADO SE LEE EL VALOR Y SE GUARDA EN B PARA LUEGO
COMPARARLO
```

```
        ACALL LEER_ADC

        MOV A, B
        CLR C
        SUBB A, #20d
        JC ES3_40
        MOV A, B
        CLR C
        SUBB A, #40d
        JNC ES3_80
        AJMP ES3_60
ES3_40:                         ;LA TEMPERATURA ES MENOS A 40
        MOV EVENTO, #1
        RET
ES3_60:                         ;LA TEMPERATURA ESTA ENTRE 40 Y 80
        MOV VALOR_DISPLAY, #11
        ACALL UPDATE_DISPLAY
        ACALL ENCENDER_ADC
        RET
ES3_80:                         ;LA TEMPERATURA ES MAYOR A 80
        MOV VALOR_DISPLAY, #12
        ACALL UPDATE_DISPLAY
        ACALL ENCENDER_ADC
        RET
;=============================================LED=============================================;
ENCENDER_LED:                   ;ENCIENDE EL LED
        SETB LED
        RET
APAGAR_LED:                     ;APAGA EL LED
        CLR LED
        RET
;=============================================PWM=============================================;
ENCENDER_PWM:                   ;ENCIENDE EL PWM
        MOV PWMP, #4d
        RET
APAGAR_PWM:                     ;APAGA EL PWM
        MOV PWMP, #0
        MOV PWM0, #255
        RET
;
DECREMENTAR_FOGON:
        DEC VALOR_DISPLAY
        ACALL UPDATE_DISPLAY
        RET
INCREMENTAR_FOGON:
        INC VALOR_DISPLAY
        ACALL UPDATE_DISPLAY
        RET
;=============================================DISPLAY=============================================;
ENCENDER_PARPADEO:              ;ACTIVA LA FUNCION DE PARPADEO
        MOV CONT_PARPADEO, #0
        SETB PARPADEO           ;ACTIVA LA FLAG DE PARPADEO
        RET
APAGAR_PARPADEO:                ;DESACTIVA LA FUNCION DE PARPADEO
        MOV CONT_PARPADEO, #0
        CLR PARPADEO            ;DESACTIVA LA FLAG DE PARPADEO
        RET
ESTADO_PARPADEO:                ;NOS DICE SI LA FLAG DE PARPADEO ESTA ACTIVA
        MOV C, PARPADEO
        JC PARPADEO_500MS
        RET
PARPADEO_500MS:                 ;MIRA SI HAN PASADO 500MS DESDE EL ULTIMO PARPADEO
        ;HAN PASADO 500MS?
        INC CONT_PARPADEO
        MOV A, #5
        CLR C
        SUBB A, CONT_PARPADEO
        JZ UPDATE_PARPADEO
        RET
UPDATE_PARPADEO:                ;SI HAN PASADO 500MS CAMBIA EL ESTADO DEL DISPLAY
        MOV CONT_PARPADEO, #0
        MOV A, P0
        ANL A, #01111111b
        JNZ APAGAR_DISPLAY
```

```
        JZ ENCENDER_DISPLAY
        RET
APAGAR_DISPLAY:                 ;APAGA EL DISPLAY
        ANL P0, #0000000b
        RET
ENCENDER_DISPLAY:               ;ENCIENDE EL DISPLAY CARGANDO EL VALOR DE LA VARIABLE
                                ;VALOR DISPLAY
        ACALL UPDATE_DISPLAY
        RET
UPDATE_DISPLAY:                 ;ACTUALIZA EL VALOR DEL DISPLAY Y DEL PWM!!!
        ANL P0, #00000000b
        MOV A, VALOR_DISPLAY
        RL A
        MOV DPTR, #DISPLAY_FUNC
        JMP @A+DPTR
DISPLAY_FUNC:
        AJMP DISPLAY_0  ;0
        AJMP DISPLAY_1  ;1
        AJMP DISPLAY_2  ;2
        AJMP DISPLAY_3  ;3
        AJMP DISPLAY_4  ;4
        AJMP DISPLAY_5  ;5
        AJMP DISPLAY_6  ;6
        AJMP DISPLAY_7  ;7
        AJMP DISPLAY_8  ;8
        AJMP DISPLAY_9  ;9
        AJMP DISPLAY_P  ;10
        AJMP DISPLAY_H  ;11
        AJMP DISPLAY_HH ;12
DISPLAY_0:      ;*gfedcba
        ORL P0, #00111111b
        MOV PWM0, #255          ;0%
        RET
DISPLAY_1:
        ORL P0, #00000110b
        MOV PWM0, #229          ;10%
        RET
DISPLAY_2:
        ORL P0, #01011011b
        MOV PWM0, #204          ;20%
        RET
DISPLAY_3:
        ORL P0, #01001111b
        MOV PWM0, #178          ;30%
        RET
DISPLAY_4:
        ORL P0, #01100110b
        MOV PWM0, #153          ;40%
        RET
DISPLAY_5:
        ORL P0, #01101101b
        MOV PWM0, 127           ;50%
        RET
DISPLAY_6:
        ORL P0, #01111101b
        MOV PWM0, #102          ;60%
        RET
DISPLAY_7:
        ORL P0, #00001111b
        MOV PWM0, #76           ;70%
        RET
DISPLAY_8:
        ORL P0, #01111111b
        MOV PWM0, #51           ;80%
        RET
DISPLAY_9:
        ORL P0, #01101111b
        MOV PWM0, #25           ;90%
        RET
DISPLAY_P:
        ORL P0, #01110011b
        MOV PWM0, #0            ;100%
        RET
DISPLAY_H:
```

```
        ORL P0, #01110100b
        RET
DISPLAY_HH:
        ORL P0, #01110110b
        RET
;=========================================ZUMBADOR=========================================;
ENCENDER_ZUMBADOR:              ;ENCIENDE EL ZUMBADO A NIVEL BAJO!
        MOV CONT_ZUMB, #0
        CLR ZUMBADOR
        RET
APAGAR_ZUMBADOR:                ;APAGA EL ZUMBADOR
        MOV CONT_ZUMB, #0
        SETB ZUMBADOR
        RET
ZUMBADOR_ESTADO:                ;NOS DICE SI EL ZUMBADOR ESTA ENCENDIDO O NO
        MOV C, ZUMBADOR
        JNC ZUMBADOR_200MS
        RET
ZUMBADOR_200MS:                 ;COMPRUEBA SI HAN PASADO 200MS DESDE ENCENDER EL ZUMBADOR
        INC CONT_ZUMB
        MOV A, #2
        CLR C
        SUBB A, CONT_ZUMB
        JZ APAGAR_ZUMBADOR
        RET
;=========================================IDL=========================================;
ACTIVAR_IDL:                    ;ACTIVA EL MODO IDL Y LAS INTERRUPCIONES EXTERNAS
        ORL IEN0, #10000001b
        ORL PCON, #00000001b
        RET
DESACTIVAR_IDL:                 ;DESACTIVA TANTO EL IDL COMO LAS INTERRUPCIONES EXTERNAS
        PUSH PSW
        PUSH ACC
        ANL IEN0, #00000000b
        ANL PCON, #00000000b
        POP ACC
        POP PSW
        RET
;=========================================ADC=========================================;
ENCENDER_ADC:                   ;ENCIENDE EL ADC Y LO CONFIGURA
        ORL IEN0,#11000000b
        ORL ADCON, #00001000b
        RET
APAGAR_ADC:                     ;APAGA EL ADC
        ANL IEN0, #00000000b
        RET
LEER_ADC:                       ;LEE EL VALOR DEL ADC Y LO GUARDA EN B
        CLR FADC
        MOV B, ADCH
        RET
;=========================================TIMER=========================================;
ENCENDER_TIMER:                 ;ENCIENDE EL TIMER Y LE CONFIGURA EL VALOR DE AUTORECARGA
        ORL IEN0, #10000010b
        MOV TH0, #56
        MOV TL0, #56
        MOV T_CONT1, #0
        MOV T_CONT2, #0
        SETB TR0
        RET
APAGAR_TIMER:                   ;APAGA EL TIMER
        ANL IEN0, #00000000b
        CLR TR0
        RET
TIMER_FUNC:                     ;FUNCION DEL TIMER
        PUSH PSW
        PUSH ACC

        INC T_CONT1
        MOV A, #TIMER_CONT_1
        CLR C
        SUBB A, T_CONT1
        JNZ CONDTIMER

        MOV T_CONT1, #0
```

```
        INC T_CONT2
        MOV A, #TIMER_CONT_2
        CLR C
        SUBB A, T_CONT2
        JNZ CONDTIMER

        MOV T_CONT2, #0

        SETB TICK               ;HA PASADO 100ms!

        ACALL ZUMBADOR_ESTADO
        ACALL ESTADO_PARPADEO

        POP ACC
        POP PSW
        RET
CONDTIMER:                      ;SI NO SE CUMPLE UN CONTADOR DEL TIMER
        POP ACC
        POP PSW
        RET
;=======================================INTERRUPCIONES=======================================;
ORG 0x03
INTERRUPCION_EXTERNA:
        ACALL DESACTIVAR_IDL
        RETI
ORG 0x0B
INTERRUPCION_TIMER:
        ACALL TIMER_FUNC
        RETI
ORG 0x53
INTERRUPCION_ADC:
        PUSH PSW
        PUSH ACC

        SETB FADC

        POP ACC
        POP PSW
        RETI
;============================================END============================================;
END
```