UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES
A R A G Ó N

CONTROL DE TEMPERATURA PARA UNA TERMOFORMADORA

T E S I S

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO MECÁNICO-ELECTRICISTA
P R E S E N T A

ROGELIO DAVID GÓMEZ WONG

DIRECTOR DE TESIS: M. en I. MIGUEL ANGEL BAÑUELOS SAUCEDO

LABORATORIO DE ELECTRÓNICA
CCADET

MÉXICO, D.F. 2006

A mis padres,
por su apoyo y comprensión incondicionales.

ÍNDICE

I. INTRODUCCIÓN

El **termoformado** es un proceso que consiste en calentar y reblandecer una lámina plana de material termoplástico (acrílico extruido, polipropileno, poliestireno, polivinilo de cloruro, entre otros) y someterla a que adopte la configuración del molde correspondiente para así, obtener un producto casi terminado con una morfología particular. El proceso se usa ampliamente en la industria del empaque de productos de consumo (charolas, vasos, contenedores de hot dogs, de huevo, etc.), en el transporte (asientos, respaldos, parabrisas, spoilers, etc.), en artículos para el hogar como gabinetes de televisión, revestimientos de refrigeradores, etc.; entre otras ramas.

El termoformado consta de dos pasos principales: calentamiento y formado. La duración del ciclo de calentamiento necesita ser suficiente para ablandar la lámina, es decir, para que alcance su temperatura de trabajo, dependiendo del polímero, su espesor y su color. Los métodos de formado pueden clasificarse en:

Termoformado al vacío. El principio básico del proceso de formado al vacío es el contar con una lámina termoplástica reblandecida en un molde sellado y donde el aire atrapado será evacuado por la fuerza de vacío o succión. A medida que el aire es evacuado del molde, causa una presión negativa sobre la superficie de la hoja y por lo tanto, la presión atmosférica natural cederá para forzar a la hoja calentada a ocupar los espacios vacíos.

Termoformado a presión. Involucra presión positiva para forzar al plástico caliente dentro de la cavidad del molde, también se le conoce como *formado por soplado*; su ventaja sobre el formado al vacío radica en que se pueden desarrollar presiones más altas. El proceso es similar al anterior, la diferencia es que la lámina se presiona desde arriba hacia la cavidad del molde.

Termoformado mecánico. El proceso de termoformado no está limitado a las técnicas neumáticas; son varias las fuerzas mecánicas que se pueden aplicar. En esta técnica de moldeo, una hoja calentada es formada entre dos moldes opuestos entre sí pero con contornos similares (macho-hembra). Cuando los moldes se unen, los contornos forzarán a la hoja a tomar idéntica forma, entre el espacio creado por los dos moldes. Cualquier protuberancia en el molde macho, mecánicamente forzará al plástico en la contraparte (molde hembra).

Una **máquina termoformadora** se encarga precisamente de aplicar un proceso de termoformado a un polímero termoplástico.

Ahora, un problema de instrumentación y control relevante, específicamente en el ámbito industrial, es **controlar la temperatura**.

La industrialización creciente y la fuerte competencia exigen productos manufacturados progresivamente más uniformes, con el fin de reducir el costo

(evitando rechazos) y elevar la producción al nivel que el mercado exige. En este contexto es forzoso medir y controlar las variables que influyen en el proceso, para garantizar que la fabricación final cumplirá las normas de calidad con la mínima incertidumbre.

La temperatura es una variable preponderante en gran número de procesos industriales, y en el termoformado no es la excepción.

Así, en el proceso de termoformado de un material termoplástico (acrílico por ejemplo), la temperatura es el factor más importante; por lo que ésta deberá controlarse cuidadosamente. Bajas temperaturas ocasionan esfuerzos internos excesivos en la pieza termoformada que, posteriormente, con cambios bruscos en la temperatura ambiente disminuyen su resistencia tornándose susceptible a la deformación (fisuras) o craqueo. Altas temperaturas, por su parte, pueden provocar que el material se ampolle (hierva) y se produzcan burbujas, reduciendo su resistencia al rasgado durante el formado, también pueden producirse marcas de molde.

Ejemplos como éste demuestran la importancia de contar con controles confiables de temperatura.

I.1 DEFINICIÓN DEL PROBLEMA

En el Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET) de la UNAM, específicamente, en el Laboratorio de Electrónica, se realizan diferentes instrumentos electrónicos como fuentes de poder, generadores de funciones, contadores universales, multímetros, etc.

Una parte imprescindible del desarrollo de un instrumento electrónico, como los mencionados, es el diseño y fabricación del gabinete que contendrá las tarjetas electrónicas y demás componentes. Como una alternativa económica para la creación de gabinetes se cuenta con el proceso de termoformado.

En el Laboratorio de Ingeniería de Producto, dentro del mismo CCADET, se ha desarrollado un prototipo de máquina de termoformado al vacío (figura 1). El diseño actual es completamente manual y no posee ningún sistema de control de temperatura; es decir, los gabinetes se realizan de manera subjetiva. No existe un control sistemático de la temperatura de trabajo del material termoplástico ni del tiempo, necesario para alcanzarla.

Figura 1. Termoformadora.

Por lo tanto, en el presente trabajo se expone el desarrollo de un **sistema de control de temperatura para una máquina de termoformado basado en un microcontrolador**. El microcontrolador a utilizar es el microcontrolador PIC (Peripheral Interface Controller) modelo PIC16F877, de la compañía Microchip.

Es necesario recordar que un microcontrolador es un circuito integrado de alta escala de integración que incorpora la mayor parte de los elementos que

configuran un controlador. Su reducido tamaño minimiza el número de componentes y el costo.

Los productos que para su regulación incorporan un microcontrolador disponen de las siguientes ventajas además de los mencionados en el párrafo anterior: aumento de prestaciones; aumento de la fiabilidad, al reemplazar el microcontrolador a un elevado número de elementos, disminuye el riesgo de averías y se requieren menos calibraciones; reducción de tamaño en el producto acabado, la integración del microcontrolador en un chip disminuye el volumen; mayor flexibilidad, dado que las características de control están programadas, su modificación sólo precisa cambios en el programa de instrucciones, esto supone una importante adaptación a las circunstancias que rodean al producto final junto a una gran rapidez en la implementación de posibles cambios.

Con base a lo expuesto arriba, se decidió entonces utilizar un microcontrolador como controlador, específicamente el PIC16F877. Se eligió este modelo por su sencillez y utilidad en comparación con los modelos comerciales de otros fabricantes; además de la enorme información y material de apoyo que existe de él. Otro factor determinante en la elección de este microcontrolador es su bajo costo.

I.2 OBJETIVOS

El objetivo de ejercer un control de temperatura sobre la máquina de termoformado es el de dar certidumbre al personal del Laboratorio de Ingeniería de Producto al momento de manejar dicho aparato; cambiando la toma de decisión “al tanteo” por la toma de decisión sobre la base de un monitoreo sistemático del proceso.

Además de que la elaboración, fabricación del empaque de los instrumentos diversos que se realizan en el Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADET), sea más eficiente y rápida. Eficiente por que la cantidad de material empleado se controla mejor y sólo se utiliza el necesario lo que redunda en menores costos para el Centro y por ende para la Universidad; y rápida, ya que el proyecto de control permite implementar un proceso sistemático y desechar el método de “al tanteo”, optimizando así el tiempo.

El control de temperatura para la máquina de termoformado tendrá las siguientes características: un rango de temperatura de 0 a 240 °C, un intervalo de tiempo de 0 a 10 minutos. A través de un teclado, se seleccionará la temperatura a la que se desea se encuentre el horno de la máquina de termoformado así como, el tiempo que dicho horno permanecerá encendido a la temperatura requerida. Ambos, la temperatura y el tiempo deseado, se visualizarán en una pantalla de cristal líquido (LCD).

Un objetivo más será el terminar el sistema de control hasta su empaque: se diseñará el circuito impreso del control y se colocará éste en un gabinete, es decir, se realizará físicamente. Así como también, cuidar que dicho sistema sea económico.

II. PRINCIPIOS Y CONCEPTOS BÁSICOS

II.1 SISTEMAS DE CONTROL

Una definición muy empleada de **sistema** es la siguiente: "Un sistema es un conjunto de elementos interrelacionados que realizan una tarea común". Como ejemplos de sistemas se puede mencionar un motor eléctrico, un automóvil, etc. Un sistema se puede visualizar como un conjunto de elementos o subsistemas, los cuales interactúan entre sí, reciben interacción desde el exterior del sistema e interactúan hacia el exterior.

Un **sistema de control**, por su parte, es aquel en el cual podemos manipular o ajustar a voluntad alguna característica o propiedad. Dicha característica o propiedad no es otra cosa que alguna variable física, por ejemplo: posición, velocidad, aceleración, presión, gasto, temperatura, nivel; por mencionar las más utilizadas. En nuestro caso, se trata de la temperatura de una máquina de termoformado.

Los sistemas de control, regulan la temperatura en los calentadores de nuestras casas, en los refrigeradores, y encienden el alumbrado público. También afectan la producción de bienes y servicios ya que los encontramos en plantas de papel, plantas químicas, refinerías, plantas generadoras de electricidad, plantas de tratamiento de agua, etc. En fin, los sistemas de control nos rodean e incluso se encuentran dentro de nosotros mismos.

Los sistemas de control se clasifican en dos tipos: retroalimentados y no-retroalimentados.

Los sistemas de control no-retroalimentados, o de malla abierta, como también se les conoce, están formados por el controlador y la planta conectados en cascada (figura 2).

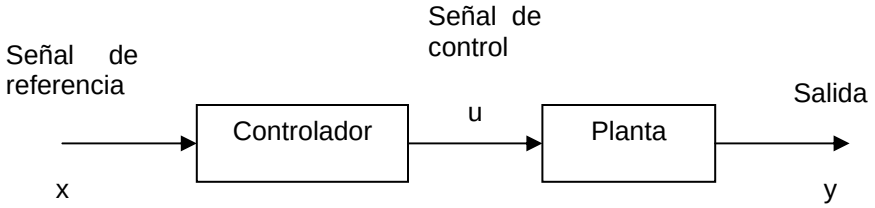

Figura 2. Sistema de control en malla abierta.

En un sistema de este tipo, no se tiene la capacidad de verificar que se obtenga el resultado deseado.

Ahora, un **sistema de control retroalimentado o de malla cerrada** se representa por medio del diagrama de bloques de la figura 3. En este caso, el

sistema cuenta con un sensor, el cual sirve para medir el valor de la salida y con ello determinar si se consiguió el resultado deseado. La señal que se obtiene del sensor se compara con la señal de referencia y se determina la señal de error.

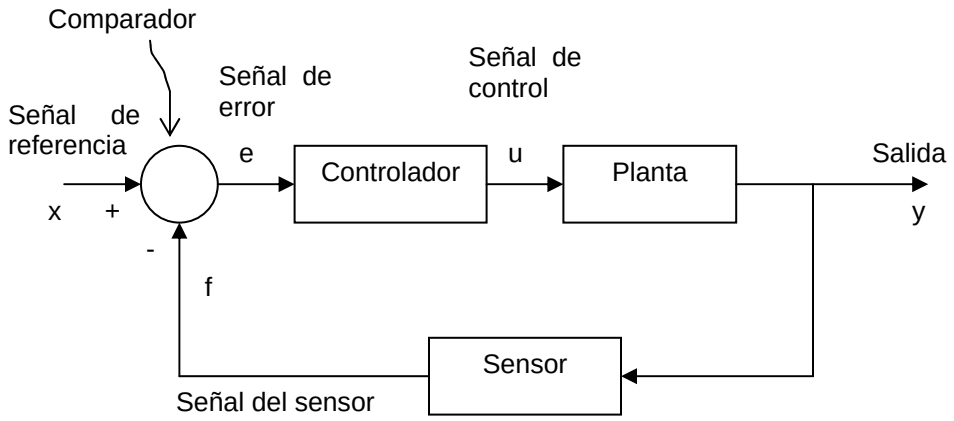

Figura 3. Sistema de control en malla cerrada.

El **controlador** recibe la señal de error y genera una señal de control cuyo objetivo será reducir el error, o bien, mantenerlo en cero. La generación de esta señal de control involucra una "decisión" por parte del controlador, y esta decisión se toma con base a una estrategia interconstruida en el controlador. Los controladores se clasifican de acuerdo al tipo de estrategia que siguen para reducir el error. Los tipos de controladores más comunes son:

Controlador ON-OFF. Es el más simple. Se denomina ON-OFF porque su salida sólo toma dos valores: encendido o apagado. Para ello simplemente determina cual es el signo del error y sobre esa base se genera la salida.

Controlador ON-OFF con histéresis. Con la finalidad de evitar la alta frecuencia de conmutación que se puede producir en los actuadores con un control ON-OFF, se realiza una modificación para proporcionarles un efecto de histéresis. El controlador ON-OFF con histéresis tiene ahora dos umbrales en lugar de uno solo. Esto le permite separar las condiciones de encendido y apagado de manera tal que no se produzcan las excesivas oscilaciones encontradas en los controladores ON-OFF normales.

Controlador proporcional. Es un controlador lineal, a diferencia de los dos anteriores. En este tipo de controlador, la acción de control es directamente proporcional al error. Se caracteriza porque al aumentar la constante de la acción proporcional K_p aumenta la velocidad de respuesta del sistema; al aumentar K_p se disminuye el error en estado estacionario; si se aumenta demasiado K_p se puede volver inestable el sistema; y es incapaz de mantener el error en cero.

Controlador proporcional-integral. Para poder eliminar el error en estado estacionario (offset) producido por los controladores proporcionales, se añade un elemento de acción integral. La acción integral permite que el error en estado estacionario se vuelva cero, sin embargo, presenta el inconveniente de volver más lenta la respuesta del sistema. Esto se debe a que al integrador le toma tiempo integrar la señal de error para producir la respuesta correcta capaz de eliminar el error. La velocidad de integración depende de la constante T_i (constante de tiempo de la acción integral), la cual está dada en segundos. Una constante muy grande hará más lento al sistema, pero una muy pequeña puede no ser suficiente para eliminar el error.

Controlador proporcional-integral-derivativo. La acción derivativa permite aumentar la velocidad de respuesta, por lo que aunada a las acciones proporcional e integral genera un controlador rápido y con cero error en estado estacionario. El modo derivativo no afecta el error en estado estable, pero proporciona una corrección rápida basado en la tasa del error; esto acelera la acción del controlador y compensa algunos de los retardos en el lazo de control. Sin embargo, amplifica el ruido de alta frecuencia, pudiendo con ello afectar a la variable controlada.

II.2 CONTROLADOR Y MICROCONTROLADOR

Como se mencionó, el controlador es el dispositivo que se emplea para el gobierno de uno o varios procesos. Aunque el concepto de controlador ha permanecido invariable, su implementación física ha variado frecuentemente. Hace tres décadas, los controladores se construían exclusivamente con componentes de lógica discreta; posteriormente se emplearon los microprocesadores, que se rodeaban con chips de memoria y E/S sobre una tarjeta de circuito impreso. Actualmente, todos los elementos del controlador se han podido incluir en un chip, el cual recibe el nombre de **microcontrolador**.

Un microcontrolador es un circuito integrado de alta escala de integración que incorpora la mayor parte de los elementos que configuran un controlador. Su reducido tamaño minimiza el número de componentes y el costo. Un microcontrolador dispone normalmente de los siguientes componentes: una Unidad Central de Proceso, memoria RAM para contener los datos, memoria para el programa tipo ROM/PROM/EPROM/EEPROM u otro, líneas de E/S para comunicarse con el exterior, diversos módulos para el control de periféricos (Temporizadores, puertos Serie y Paralelo, ADC, DAC, etc.), generador de impulsos de reloj que sincronizan el funcionamiento de todo el sistema.

Según el modelo de microcontrolador que se trate, el tamaño y tipo de memoria puede diferir, así como el número de líneas de E/S y los módulos de control de periféricos. La diversificación de modelos permite seleccionar el más adecuado según la aplicación que se trate.

En el presente trabajo, se expone el desarrollo de un **sistema de control de temperatura para una máquina de termoformado basado en un microcontrolador**.

Se decidió utilizar como controlador un microcontrolador, específicamente el **microcontrolador PIC (Peripheral Interface Controller) modelo PIC16F877** de la compañía Microchip, por las razones expuestas en el capítulo anterior además de las siguientes: velocidad de ejecución y eficiencia en la compactación del código, gracias a la arquitectura Harvard y la técnica de segmentación (pipe-line) que posee; así, como a la memoria FLASH que integra.

La memoria FLASH se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar. Funciona como una ROM y una RAM pero consume menos y es más pequeña. A diferencia de la ROM, la memoria FLASH es programable en el circuito. Es más rápida y de mayor densidad que la EEPROM y tolera más ciclos de escritura/borrado.

Además, el microcontrolador PIC16F877 cuenta con un convertidor analógico-digital de 10 bits; así como también, con la capacidad de comunicarse con una computadora por vía serial (módulo USART), lo cual abre la posibilidad de que en un futuro el sistema de control se maneje a través de ella.

II.3 CONTROL DE TEMPERATURA

El sistema propuesto consiste de un sensor de temperatura, un microcontrolador, un teclado, un display de cristal líquido (LCD) y una etapa de potencia para accionar la resistencia eléctrica de la termoformadora. Ver figura 4.

A través del teclado, el usuario indica la temperatura de operación deseada del horno de la máquina de termoformado (en el rango de 0 a 240 °C), así como el tiempo que se mantendrá el horno a dicha temperatura (en el intervalo de 0 a 10 minutos). El display de cristal líquido permite visualizar la entrada de datos (la temperatura y el tiempo requeridos) como también, la temperatura actual del horno y el tiempo transcurrido.

El microcontrolador utiliza la temperatura deseada (referencia) y la temperatura medida (señal del sensor) para generar una señal de control de tipo ON-OFF. La salida (señal de control) se envía a través de una de las líneas de E/S del microcontrolador hacia la etapa de potencia. Dicha etapa se encuentra separada del resto del circuito electrónico mediante un optoacoplador, el cual activa un triac con capacidad para manejar 40 A.

Una vez transcurrido la temperatura y el tiempo especificados, el usuario vía teclado abre la electroválvula de la bomba de vacío de la termoformadora, lo cual provoca que la lámina de material termoplástico adquiera la forma del molde correspondiente; posteriormente, se cierra la electroválvula también vía teclado.

A continuación, se presenta el diagrama a bloques del sistema de control de temperatura para la termoformadora:

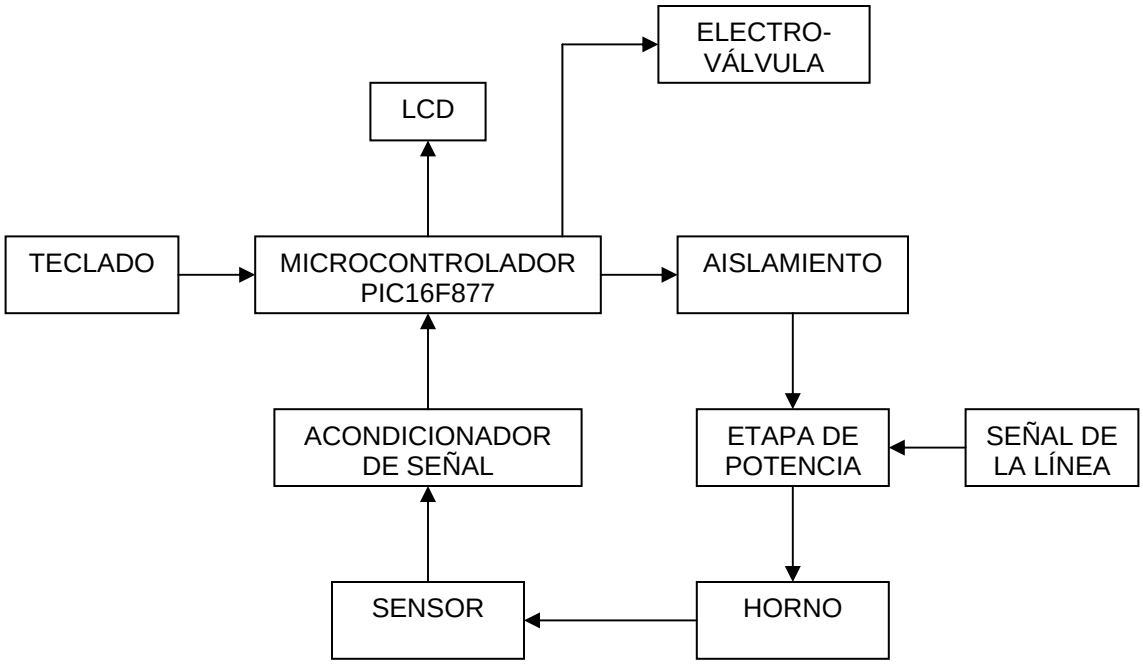

Figura 4. Diagrama a bloques del sistema propuesto.

III. SENSOR-ACONDICIONADOR DE SEÑAL

III.1 TECNOLOGÍAS DE SENSADO DE TEMPERATURA

Hoy en día, los sensores de temperatura más utilizados son: el **termopar**, el RTD (Resistance Temperature Detector), el termistor y el sensor en circuito integrado. El rango de temperatura, la sensibilidad, la exactitud, el costo, la linealidad así como el fácil diseño de circuitos de soporte, son algunas de las características que hay que tener en cuenta al momento de seleccionar un sensor de temperatura, según los requerimientos de la aplicación. Ningún sensor es el idóneo para todas las situaciones.

RTDs. Los RTD son conductores, de Platino, Níquel o Cobre, que presentan una resistencia que varía con la temperatura; su coeficiente de temperatura es positivo, al aumentar la temperatura del conductor aumenta su resistencia. La curva de resistencia-temperatura de este tipo de sensor es casi lineal; es decir, los sensores RTD poseen una excelente linealidad. Su rango de temperatura va de –250 a 900 °C.

Sin embargo, tienen ciertos inconvenientes: son susceptibles al daño (deformaciones mecánicas) como resultado de vibraciones, esto es por que sus conductores son calibre 26/30 AWG lo cual los hace propensos a la fractura. Y además, presentan autocalentamiento debido a la fuente de corriente que requieren: la fuente de corriente permite al RTD convertir la resistencia en voltaje, el cual debe ser de un valor tal que no se vea afectado por los niveles de ruido; entonces, se debe hacer pasar un valor alto de corriente a través del elemento RTD. Pero lo anterior, provoca que la temperatura del RTD se incremente y esto a su vez incrementa el valor de su resistencia lo que afecta la exactitud de la medición.

Termistores. Los termistores también son resistencias variables dependientes de la temperatura pero, no están basados en conductores como los RTD sino en semiconductores; por lo que su coeficiente de temperatura es negativo, al aumentar la temperatura disminuye su resistencia. El rango de temperatura de este tipo de sensores es de –100 a 450 °C. El cambio en la resistencia al variar la temperatura es mayor en el termistor que en el RTD, es decir, el termistor es más sensible. Esta alta sensibilidad del termistor redunda en una gran exactitud en la medición.

Pero, al graficar la resistencia del termistor contra la temperatura se obtiene una función no lineal, una curva; el termistor es pues, menos lineal que un RTD, de tal forma que se requiere de un polinomio de tercer orden para su linealización. Además, al igual que los RTD, los termistores presentan autocalentamiento, sobre todo a temperaturas altas donde sus resistencias son muy bajas.

Sensores de temperatura en circuito integrado. Este tipo de sensores se distingue por su funcionalidad: por ser un circuito integrado, pueden incluir circuitos de

procesamiento de señales. No se necesita diseñar circuitos de compensación o linealización para los sensores de temperatura en circuito integrado; así como tampoco diseñar comparadores o circuitos ADC para convertir las salidas analógicas en niveles lógicos o códigos digitales. Todas estas funciones ya vienen incorporadas en varios modelos.

Las desventajas de los sensores de temperatura en circuito integrado al compararseles con los otros tipos de sensores son: el rango de temperatura, que va de los –55 °C a los 150 °C; y la exactitud, que es de ±1 °C.

III.2 EL TERMOPAR

El termopar consiste de dos alambres de diferentes metales que están unidos en un extremo (figura 5). Dicha unión (T1) se localiza donde se quiere medir la temperatura. Si existe una diferencia de temperatura entre el punto de juntura (T1) y el otro extremo de los alambres (T2), un voltaje aparecerá entre los dos alambres en el extremo donde éstos no están soldados. A este voltaje se le denomina comúnmente voltaje EMF (thermocouple´s Electromotive Force), el cual cambia con la temperatura sin necesidad de usar una fuente de corriente o voltaje como excitación. Si la diferencia de temperatura entre los dos extremos del termopar cambia, el voltaje EMF cambiará también.

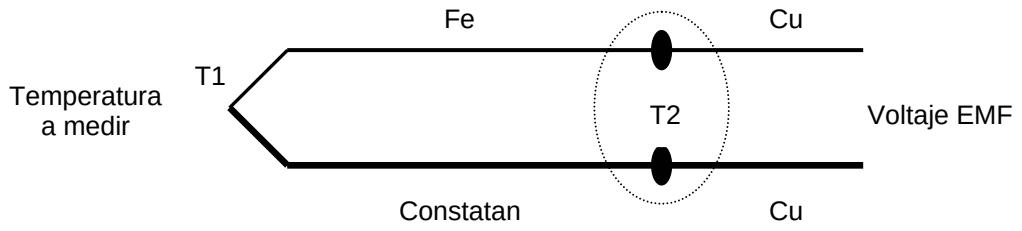

Figura 5. Esquema de un termopar.

Hay tantas variedades de termopares como metales, sin embargo, algunas combinaciones trabajan mejor que otras. En la Tabla 1 se muestra la lista de termopares más usados en la industria, cuyos comportamientos han sido estandarizados por la NIST (National Institute of Standards and Technology).

Este tipo de sensores de temperatura presenta algunas ventajas con respecto a los otros sensores (RTD, termistor, sensor en circuito integrado). Como ya se dijo, el termopar no requiere de ninguna excitación eléctrica, como una fuente de voltaje o corriente; de hecho, un intento por proveer de alimentación externa podría introducir errores al sistema.

Termopar	Material	Rango de Temperatura (°C)	Coeficiente de Seebeck (@ 20°C)
E	Cromel (+) Constatan (-)	-200 a 900	62μV/°C
J	Hierro (+) Constatan (-)	0 a 760	51μV/°C
T	Cobre (+) Constatan (-)	-200 a 371	40μV/°C
K	Cromel (+) Alumen (-)	-200 a 1260	40μV/°C
N	Nicrosil (+) Nisil (-)	0 a 1260	27μV/°C
B	Platino (30% Rodio) (+) Platino (6% Rodio) (-)	0 a 1820	1μV/°C
S	Platino (10% Rodio) (+) Platino (-)	0 a 1480	7μV/°C
R	Platino (13% Rodio) (+) Platino (-)	0 a 1480	7μV/°C

Tabla 1

El costo de los termopares varía dependiendo de la pureza de los metales, la integridad de la juntura y la calidad del aislamiento del alambre. A pesar de ello, los termopares son relativamente baratos comparados con el resto de los sensores de temperatura.

El termopar es uno de los pocos sensores que puede soportar ambientes hostiles. Capaz de mantener su integridad en un amplio rango de temperatura, así como también, soportar atmósferas corrosivas o tóxicas. Además, es resistente al manejo rudo. Esto es sobre todo una consecuencia del uso de galgas de alambre más pesadas en la construcción de los termopares. Adicionalmente, los aislamientos usados realzan la robustez del termopar.

Sin embargo, los termopares también presentan desventajas. El voltaje que producen es muy pequeño, del orden de decenas de microvolts por grado centígrado, y no lineal. En la figura 6 se ilustra la no-linealidad del termopar, con la primera derivada del voltaje EMF contra la temperatura. La primera derivada en una temperatura especificada se denomina el Coeficiente de Seebeck. El Coeficiente de Seebeck es una estimación linealizada de la deriva de la temperatura del punto de unión del termopar sobre un pequeño rango de temperatura. Debido a que todos los termopares son no lineales, el valor de este coeficiente cambia con la temperatura especificada.

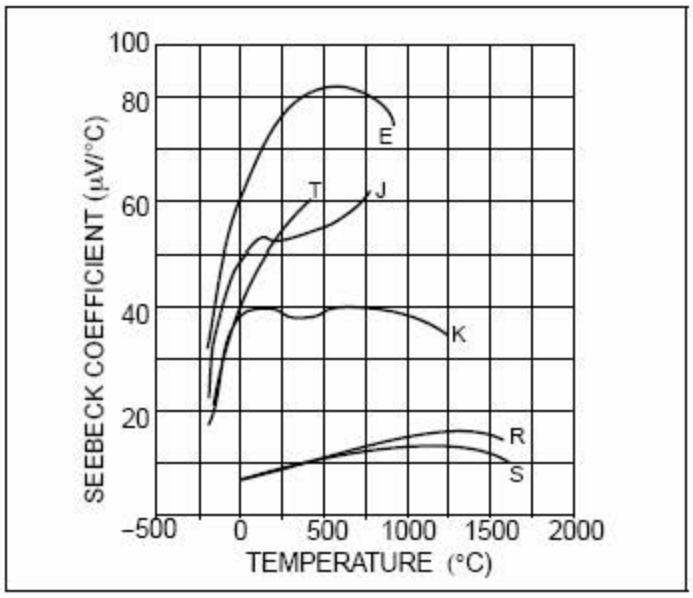

Figura 6. Coeficiente de Seebeck de varios termopares contra la temperatura.

El hecho de que la salida del termopar sea muy pequeña, implica que a la porción de acondicionamiento de la señal es necesario incluirle una etapa de ganancia. También, por ser no lineal el voltaje generado, se requieren de métodos de linealización, ya sea por hardware o software.

Al extremo T2 en la figura 5, se le conoce como Unión de Referencia o Punto de Compensación de Unión Fría. En dicho punto se crean otro par de termopares (Hierrro-Cobre y Constatan-Cobre), los cuales introducen un error significativo al sistema. Para eliminar dicho error es necesario sensar la temperatura en la Unión de Referencia, por lo que parte del circuito de acondicionamiento de la señal debe ser dedicado a realizar dicha actividad.

Así pues, de los sensores de temperatura mencionados podemos decir que, el termopar es el más apropiado cuando se desea sensar altas temperaturas, mientras el RTD es el mejor para bajas temperaturas donde se requiere una buena linealidad. El termistor se usa típicamente en aplicaciones con rangos pequeños de temperatura, pero ofrece una mayor exactitud que el termopar o el RTD. El sensor en circuito integrado por su parte, es fácil de usar.

Ahora, para el control de temperatura de la termoformadora se elige el termopar. Las razones para tal elección son las siguientes:

El termopar no requiere de una alimentación externa (voltaje o corriente), por lo que no es necesario diseñar un circuito para excitar al sensor lo que redunda en un costo menor.

Por su amplio rango de temperatura. De los materiales termoplásticos, el policarbonato tiene la mayor temperatura de termoformado (230 °C), motivo por el cual el rango de temperatura del sistema de control es de 0 a 240 °C, diez grados centígrados más por seguridad.

Este rango puede ser cubierto por un RTD o un termistor, pero se usa un termopar por que no presenta autocalentamiento; lo cual es gracias a su baja masa termal, lo que también se refleja en una mayor velocidad de respuesta.

A pesar de que un RTD se caracteriza por su linealidad y un termistor por su exactitud, ambos son "delicados" comparados con el termopar; no soportan ambientes hostiles ni un manejo rudo como es el caso del termopar.

III.3 EL TERMOPAR TIPO K

El termopar puede ser fabricado de cualquier par de metales, sin embargo, en la práctica las combinaciones se hacen considerando que tengan una resistencia adecuada a la corrosión, a la oxidación, a la reducción y a la cristalización; que desarrollen un voltaje relativamente alto, que sean estables, de bajo costo, de baja resistencia eléctrica y que la relación entre la temperatura y el voltaje sea tal que el aumento de éste sea (aproximadamente) proporcional al aumento de temperatura. En la Tabla 1 se muestra un resumen de las combinaciones más utilizadas.

En nuestro caso, se emplea un **termopar tipo K**. Con base en la Tabla 1, este termopar tiene una composición de Cromel-Alumen, y un margen de medida de –200 a 1260 °C.

Se escoge este tipo de termopar a pesar de que su Coeficiente de Seebeck no es el mejor (40μV/°C) comparado con el tipo J (51μV/°C) o el tipo E (62μV/°C); incluso es igual al del tipo T. La razón estriba en el hecho de que la variación de su Coeficiente de Seebeck con la temperatura es mucho más "estable" que la del resto de los termopares, como podemos apreciar en la figura 6, para el rango de temperatura que nos interesa (0 a 250 °C).

Esta "estabilidad" es importante, ya que el Coeficiente de Seebeck es usado cuando se diseña la porción del hardware del sistema que sensa la temperatura en la Unión de Referencia; entonces, el hecho de que el coeficiente permanezca "estable" disminuye la complejidad del circuito, así como también, el grado de linealización que se requiere.

III.4 ACONDICIONADOR DE SEÑAL

Las desventajas que presentan los termopares son: un bajo nivel de salida, una pobre sensibilidad y una no-linealidad muy alta comparada con la de los demás tipos de sensores; además de necesitar de una temperatura de referencia. Esto hace que el circuito acondicionador de señal incluya una etapa de ganancia, una de compensación y otra de linealización.

Retomando la figura 5, se mencionó que en el extremo T2 se crean otro par de termopares, los cuales introducen un error significativo al sistema. Para eliminar dicho error se requiere hacer a T2 independiente de las variaciones de temperatura del ambiente, una práctica común para lograrlo es mantener en 0 °C a T2 mediante su inmersión en una mezcla de hielo-agua; razón por la cual a dicho extremo se le denomina Punto de Compensación de Unión Fría o Unión de Referencia. Sin embargo, dicha solución a pesar de ser de una gran exactitud es de difícil mantenimiento y muy costosa. Por lo tanto, se emplea una unión fría simulada electrónicamente es decir, una compensación electrónica de la Unión de Referencia.

La idea consiste básicamente, en dejar que la Unión de Referencia sufra las variaciones de temperatura del ambiente, pero sumar un voltaje dependiente de la temperatura al voltaje EMF tal que el voltaje que se obtiene sea el mismo como si el extremo T2 estuviera a 0 °C. Esta fuente de voltaje dependiente de la temperatura se denomina Compensador de Unión Fría. Su salida está diseñada para ser 0 V a 0 °C y tener una pendiente igual al Coeficiente de Seebeck sobre el rango esperado de temperaturas de T2.

Para operar correctamente, el Compensador de Unión Fría tiene que estar a la misma temperatura que el extremo T2; por lo que el compensador debe de estar físicamente cerca de la Unión de Referencia. Así pues, dicho compensador es un sensor de la temperatura en la Unión de Referencia.

Para el control de temperatura de la termoformadora se usa como Compensador de Unión Fría el circuito integrado LT1025.

Este dispositivo es un Compensador de Unión Fría compatible con los termopares tipo E, J, K, R, S y T; mide la temperatura ambiente (en este caso, la de la Unión Fría) y ofrece una tensión de compensación para los distintos termopares. Las tensiones de salida del LT1025 son 60.9μV/°C (E), 51.7μV/°C (J), 40.6μV/°C (K y T), y 6μV/°C (R y S), para una temperatura de 25 °C. Además de una salida auxiliar de 10mV/°C.

Como se observa en la figura 6, conforme los rangos de temperatura se incrementan, la pendiente de los termopares cambia; solo que dicho cambio en la pendiente no es constante sino más bien una cuasi-parabólica. Por lo que las salidas del LT1025 poseen un arco parabólico para ayudar a compensar este

efecto y así, mantener la compensación de la Unión Fría lo más exacta posible sobre un rango de temperatura más ancho.

La corriente que consume el LT1025 es de 80μA, resultando en un incremento de su temperatura interna menor a 0.1 °C para voltajes de alimentación menores a 10V; es decir, su autocalentamiento es mínimo.

Por su parte, para la etapa de ganancia se requiere de amplificadores operacionales con un muy bajo voltaje de offset y deriva. En este caso, se usa el OPA27, el cual es un amplificador operacional en circuito integrado de ultra bajo nivel de ruido y alta precisión con las siguientes características: un voltaje de offset de 25μV y, una deriva de 0.4μV/°C.

En la siguiente figura se muestra el circuito acondicionador de señal, el cual entrega una salida de 10mV/°C.

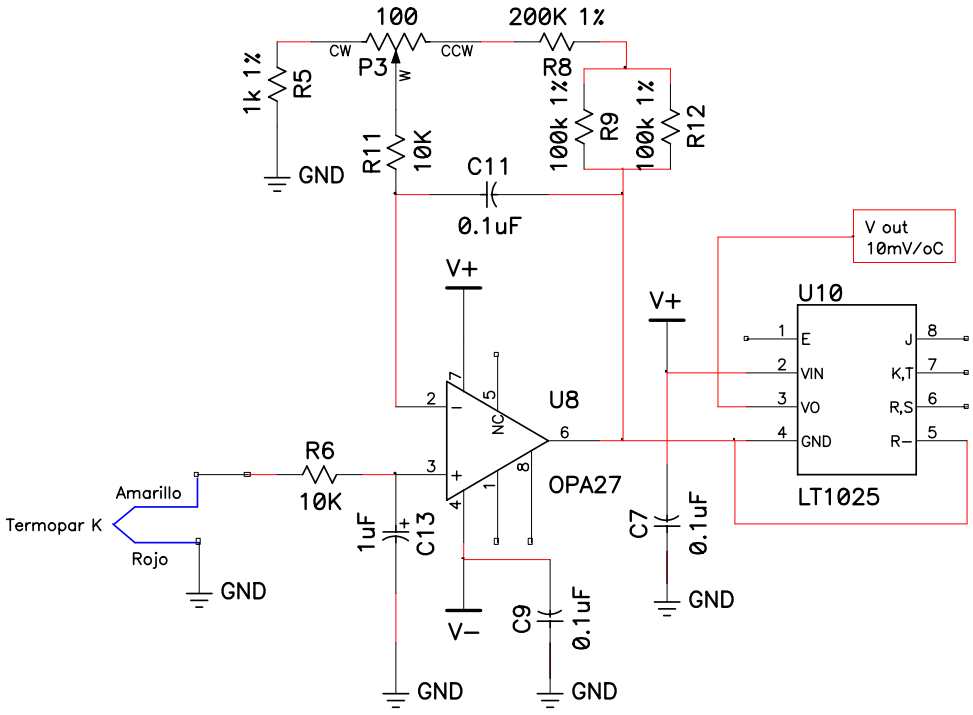

Figura 7. Circuito acondicionador de señal.

Para ajustar dicho circuito, se usa un multímetro digital de 3 ½ dígitos marca OTTO modelo MX-250T, con capacidad para lecturas de temperatura en el intervalo de –20 °C a 1370 °C con una resolución de 1 °C; un termómetro digital marca Hewlett-Packard modelo 2802A de 4 ½ dígitos, el cual posee un rango de temperatura de –200 a 600 °C, una resolución de 0.1 °C y una exactitud de ±0.5 °C; y, un multímetro marca FLUKE modelo FLUKE 45 con display dual de 5

dígitos, una resolución de 100μV y una exactitud de 0.02% + 2 para un rango de voltaje de DC de 3V.

La salida del Compensador de Unión Fría LT1025 (pin 3), en la figura 7, se conecta a la entrada de voltaje del multímetro FLUKE para visualizar el voltaje de DC que entrega el circuito acondicionador, sabiendo de antemano que tiene una tasa de 10mV/°C. Así, si se despliega un voltaje de 300mV, éste se divide por la tasa para obtener la temperatura que, en este caso, sería de 30 °C.

La calibración se realiza de la siguiente manera: primero, la punta del termopar del circuito acondicionador de señal, la del termómetro digital Hewlett-Packard y la del multímetro OTTO, se sumergen en una mezcla de hielo-agua cuya temperatura es de 0 °C; en el termómetro digital y en el multímetro OTTO aparece una temperatura de 0 °C. Entonces, el preset de 100Ω (P3) del circuito acondicionador se varía hasta que en el multímetro FLUKE se despliega una lectura de 0mV. Cuando esto sucede, el circuito acondicionador se encuentra ajustado.

Para comprobar dicho ajuste, las tres puntas se sumergen ahora en agua hirviendo. El termómetro digital marca 100 °C, al igual que el multímetro OTTO; si esta bien calibrado el circuito acondicionador, debe de marcar también, a través del multímetro FLUKE, 1000mV.

Finalmente, la linealización de la señal del termopar puede realizarse por hardware o software. Los métodos de linealización por software son: por polinomios y, por tablas de referencia. En cuanto al hardware, una técnica común es la compensación.

A pesar de que la linealización es más limitada por hardware que por software, se usa la primera aprovechando la etapa de compensación con el LT1025 y que, las salidas de éste poseen un arco parabólico. Además de que no se requiere un grado de linealización muy alto debido a la aplicación y a que, el Coeficiente de Seebeck es “constante” en el rango de temperatura de operación que se necesita.

IV. ETAPA DE CONTROL

IV.1 INTRODUCCIÓN

La etapa de control, es la parte más importante del sistema de control de temperatura propuesto, el cual, se trata de un sistema de control retroalimentado o de malla cerrada; estos sistemas, se caracterizan por tener la capacidad de verificar que se obtenga el resultado deseado. El sistema de control retroalimentado cuenta con un sensor, el cual sirve para medir el valor de la salida y con ello determinar si se consiguió el valor requerido. La señal que se obtiene del sensor, se compara con la señal de referencia y se determina la señal de error. El controlador recibe la señal de error, y genera una señal de control cuyo objetivo será reducir el error, o bien, mantenerlo en cero.

Con base en el diagrama de bloques que se muestra en la figura 3 (capítulo II, página 7), la etapa de control se encuentra constituida por el comparador y el controlador. El diagrama a bloques de dicha etapa es el siguiente:

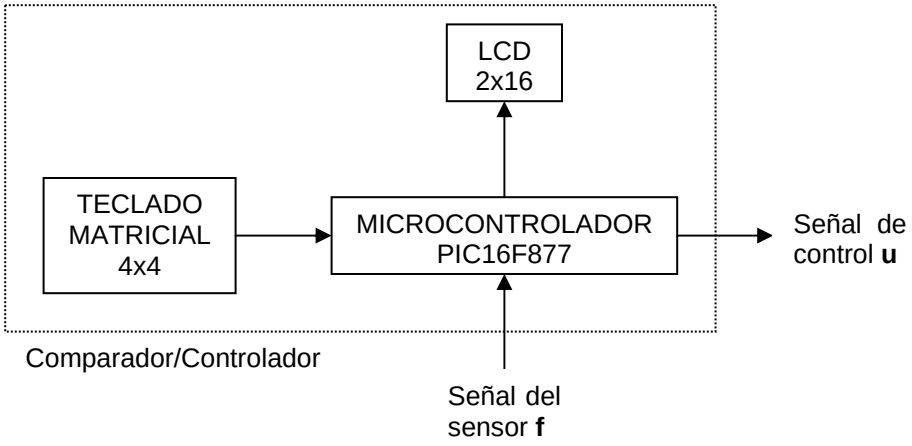

Figura 8. Diagrama a bloques de la etapa de control.

Como se observa en la figura de arriba, esta etapa incluye un teclado. A través del teclado, el usuario indica la temperatura de operación deseada del horno de la máquina de termoformado (en el rango de 0 a 240 °C), así como el tiempo que se mantendrá el horno a dicha temperatura (en el intervalo de 0 a 10 minutos); también permite abrir y cerrar la electroválvula, así como, reiniciar el proceso de termoformado o iniciar otro.

En la figura 8, también aparece una pantalla de cristal líquido (LCD). Este módulo de visualización, hace que el control de temperatura sea accesible y fácil de manejar, ya que permite observar el desarrollo del proceso: la entrada de datos (la temperatura y el tiempo requeridos), la temperatura actual del horno y el

tiempo transcurrido, la indicación de abrir y cerrar la electroválvula y, la de un nuevo proceso de termoformado.

En cuanto al microcontrolador, se trata de un PIC modelo PIC16F877 de la compañía Microchip. Representa el núcleo de la etapa de control, en él se localizan (por software), el comparador y el controlador.

El controlador, es un dispositivo que se emplea para el gobierno de uno o varios procesos; recibe la señal de error y genera las señales adecuadas (señal de control) que accionan los mecanismos adecuados para contrarrestar dicho error. La generación de esta señal de control involucra una "decisión" por parte del controlador, y esta decisión se toma con base a una estrategia interconstruida en el controlador. Los controladores se clasifican de acuerdo al tipo de estrategia que siguen para reducir el error. Los tipos de controladores más comunes son: controlador ON-OFF, controlador ON-OFF con histéresis, controlador proporcional, controlador proporcional-integral, controlador proporcional-integral-derivativo.

IV.2 CONTROL ON/OFF

El controlador más simple empleado en sistemas de control es el **controlador ON/OFF**. Se denomina ON/OFF porque su salida sólo toma dos valores: encendido o apagado. Para ello simplemente determina cual es el signo del error, y sobre esa base se genera la salida. De esta manera se tienen dos posibilidades:

- Si el error es positivo, el controlador se enciende; y si el error es negativo, se apaga.
- Si el error es negativo, el controlador se enciende; y si el error es positivo, se apaga.

En la figura 9, se muestra un diagrama de bloques de un sistema de control con un controlador ON/OFF. Por simplicidad, no aparece el bloque correspondiente al sensor, puesto que se considera con función de transferencia unitaria.

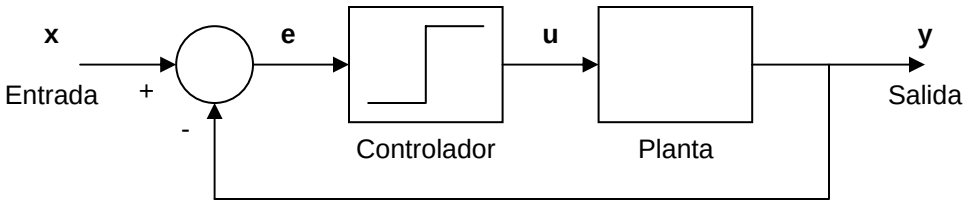

Figura 9. Controlador ON/OFF.

La relación entre el error (e) y la acción de control (u) se muestra gráficamente en la figura 10. Se observa que cuando el error es positivo, la acción de control es igual a encendido; mientras que cuando el error es negativo, la acción de control es apagado. El caso cuando el error es igual a cero no está previamente determinado, y por lo tanto podemos fijarlo a cualquiera de los dos estados. Por ejemplo si lo fijamos como encendido, la función del controlador queda definida como:

$$u = \begin{cases} \text{ON, si } e \geq 0 \\ \text{OFF, si } e < 0 \end{cases}$$

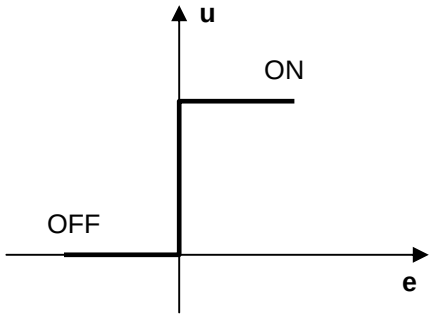

Figura 10. Gráfica entrada vs. salida del controlador ON/OFF.

Es importante mencionar, que no es posible obtener la función de transferencia del controlador ON/OFF, debido a que se trata de un sistema no-lineal. Es decir, al aplicar el doble de la señal de entrada, no se obtiene el doble de la señal de salida. Esto se puede constatar fácilmente en la gráfica de la figura 10.

En la figura 11, se muestran las gráficas de respuesta de un sistema de control ON/OFF como el de la figura 9. Supongamos que se trata de controlar la temperatura de una cafetera. El objetivo de control es elevar la temperatura del agua desde condiciones iniciales hasta 40 °C. En la gráfica A, se observa que la señal de salida va creciendo lentamente hasta alcanzar los 40 °C, mientras que la señal de error (gráfica B) va disminuyendo hasta aproximarse a cero. En la gráfica C se muestra la acción de control, es decir, la resistencia eléctrica de la cafetera encendiéndose y apagándose. Se puede ver que una vez que se alcanza la temperatura deseada, la temperatura empieza a oscilar alrededor del valor deseado y esto produce un rápido ciclo de encendido y apagado de la señal de control. Para muchos de los sistemas esto resulta en un excesivo desgaste del elemento actuador y por lo tanto es indeseable.

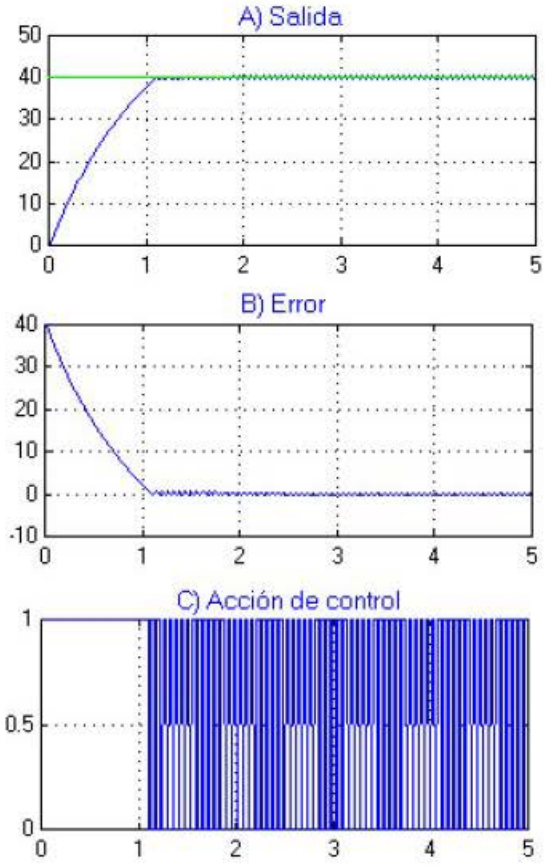

Figura 11. Gráficas de respuestas del controlador ON/OFF.

Con la finalidad de evitar la alta frecuencia de conmutación que se puede producir en los actuadores con un control ON/OFF, se realiza una modificación para proporcionales un efecto de histéresis.

El controlador ON/OFF con histéresis tiene ahora dos umbrales en lugar de uno solo. Esto le permite separar las condiciones de encendido y apagado de manera tal que no se produzcan las excesivas oscilaciones encontradas en los controladores ON/OFF normales.

Si se aumenta la distancia entre los umbrales disminuye la frecuencia de los ciclos de encendido-apagado. Sin embargo, esto presenta el inconveniente de que la variable controlada ampliará sus oscilaciones alrededor del valor deseado; por lo tanto, se debe establecer un compromiso entre las oscilaciones permitidas a la variable de salida, y su consecuente desviación del valor deseado, contra la frecuencia de los ciclos de encendido-apagado que genera la acción de control.

Como se dijo, un control ON-OFF o de dos posiciones, es el caso más simple de un control retroalimentado; este tipo de controles es muy económico en su construcción, fácil de implementar, operar y calibrar. Por lo cual, para el control de la temperatura de la termoformadora, se usa un controlador ON-OFF.

Además, se selecciona este tipo, a pesar de tener como inconveniente una alta frecuencia de conmutación, debido a que la variable física a controlar (la temperatura) cambia lentamente con el tiempo; a pesar de ello, pareciera que si se elige un controlador ON/OFF es mejor que sea con histéresis.

Si a un cuerpo le aportamos calor, éste eleva su temperatura. Si lo hace lentamente, decimos que tiene mucha capacidad calorífica, puesto que es capaz de almacenar mucho calor por cada grado centígrado de temperatura. La diferencia de capacidad calorífica entre el agua y el aceite, por ejemplo, (mayor en el primero que en el segundo) es lo que hace que, al fuego, el agua tarde más en calentarse que el aceite, pero también que el agua "guarde" más el calor.

La capacidad calorífica y el almacenamiento de calor traen aparejados ciertos fenómenos. Por ejemplo, en casa, en invierno, cuando encendemos la calefacción al llegar por la tarde, la habitación tarda en alcanzar una temperatura agradable; y cuando la apagamos por la noche, la temperatura de la habitación todavía es buena y no se enfría inmediatamente. Esto ocurre también en las estaciones en el hemisferio norte, el 21 de Abril (Equinoccio de primavera) el sol está en la misma posición que el 21 de Septiembre (Equinoccio de otoño), y sin embargo, las temperaturas son mayores en esta última fecha, por la sencilla razón de que la tierra todavía "guarda" el calor del verano, que irá perdiendo poco a poco. Esta "resistencia" de la temperatura a reaccionar de inmediato a los aportes de calor, es lo que se llama inercia térmica.

No se elige un controlador ON/OFF con histéresis porque se considera la inercia térmica de la resistencia del horno de la termoformadora, la cual produce un efecto de histéresis.

IV.3 PROGRAMA

La parte central de la etapa de control es el microcontrolador PIC modelo PIC16F877 de la compañía Microchip. En él se encuentran implementados, mediante software, el comparador y el controlador; además de las rutinas para comunicarse con el teclado y la pantalla de cristal líquido (LCD), las que generan la señal de control, las que adecuan la señal proveniente de la etapa de sensado, en fin, todo el proceso de control.

El problema de manejar un microcontrolador se reduce a saber programarlo. Se han desarrollado todo tipo de lenguajes de programación para los microcontroladores, pero los más usados son el **lenguaje Ensamblador**, el

BASIC y el lenguaje C. De éstos, se elige el Ensamblador porque los programas escritos en este lenguaje son compactos y rápidos.

El único lenguaje que sabe interpretar la Unidad Central de Proceso (CPU) del microcontrolador es el binario, por lo que en la memoria de instrucciones éstas tienen implementado su código OP en binario, es decir, con unos y ceros. A esta forma de representación se la denomina 'código objeto', y por eso los ficheros que contienen este formato tienen la extensión ∗.OBJ .

Dado lo poco inteligible y descifrable que tiene el lenguaje binario para el hombre, se descarta la posibilidad de escribir los programas ejecutables directamente. El Ensamblador, es un lenguaje de bajo nivel porque cada una de sus instrucciones se corresponden con otra capaz de interpretar el CPU. Es más humano porque en vez de unos y ceros, emplea conjuntos de letras que "recuerdan" la operación que realiza la instrucción (mnemónico), y es muy cercano a la máquina por la correspondencia que tiene con sus instrucciones.

El programa escrito en un lenguaje diferente al lenguaje máquina (como también se le denomina al lenguaje binario) recibe la denominación de 'fuente' y, cuando está escrito en Ensamblador, suele tener la extensión ∗.ASM o ∗.SRC . Se le denomina fuente debido a que dicho programa no puede ser ejecutado por el microcontrolador, no lo entiende. Debe ser traducido a código objeto en lenguaje binario de lo cual se encarga un programa traductor, el cual se denomina compilador si el lenguaje de programación es de alto nivel (por ejemplo lenguaje C) o, Ensamblador si el lenguaje es de bajo nivel (lenguaje Ensamblador).

La edición del programa de control en el lenguaje seleccionado (lenguaje Ensamblador), se realiza en un editor de textos que trabaje con caracteres ASCII, como el Bloc de Notas. En este proyecto, se usa el programa **MPLAB IDE v6.30** que distribuye en forma gratuita Microchip a través de Internet. Este software permite editar, simular, ensamblar o compilar, y grabar los programas fuente.

Editado el programa y convertido a código máquina, hay que grabarlo en la memoria de instrucciones del microcontrolador. Esto se realiza con un 'grabador', que es un dispositivo que consta de un soporte físico (zócalo) sobre el que se sujeta el circuito integrado a grabar (microcontrolador). El control del grabador se efectúa desde una computadora que, con el software adecuado (en este caso, MPLAB IDE), se encarga de escribir en la memoria no volátil el programa que requiere la aplicación. El grabador empleado es el **PICSTART PLUS** de Microchip.

A continuación, se detalla el programa para el sistema de control de la temperatura de la termoformadora, sus subrutinas, interrupciones, bucles, tratamiento de entradas y salidas.

IV.4 CÓDIGO FUENTE

Como referencia, el código fuente del programa del sistema propuesto se localiza en el Apéndice (páginas 66 a 77).

Al inicio del programa, en la cabecera, se tiene un comentario que indica el nombre del mismo (PROYECTO.ASM). Seguidamente, nos encontramos con la expresión *LIST*; esta expresión y las que veremos a continuación son 'directivas' para el Ensamblador. Una directiva, es un comando escrito en el código fuente para realizar un control directo o ahorrar tiempo a la hora de ensamblar. La directiva *LIST* dispone de diferentes opciones que permiten elegir, entre otras cosas, el tipo de procesador a emplear (P), número de caracteres por línea (C), tamaño de los tabuladores (B), etc.

RADIX por su parte, es la directiva que especifica el sistema de numeración a utilizar en los datos.

En ocasiones, se requiere más de una subrutina en el programa, siendo habitual que algunas se precisen en otros programas. En estos casos, es conveniente disponer de 'bibliotecas de subrutinas', que se denominan 'librerías'. En cada programa se cargan desde la librería las subrutinas que se necesiten. La directiva *INCLUDE* "pega" el fichero que se referencia en el programa. Dicho fichero se inserta en el código durante el proceso de ensamblado.

Las siguientes directivas que encontramos en el código fuente son las *EQU*; éstas asignan valores a las etiquetas deseadas. Así, por ejemplo, *BASE* tiene asignado el valor *0x2E* que corresponde a la dirección (en hexadecimal) del área de datos del registro *BASE*. En esta sección, se definen los registros no específicos a utilizar durante el programa.

La directiva *ORG*, le indica al Ensamblador dónde debe comenzar a colocar las instrucciones en la memoria de programa (memoria ROM), es decir, el OriGen para todo el código que sigue. La dirección de comienzo es en la posición cero, debido a que la familia de microcontroladores PIC de gama media, después del encendido o RESET siempre ejecutan la instrucción situada en la dirección cero. Se denomina 'Vector de Reset'.

La instrucción *goto INICIO* ha sido colocada en la dirección cero, que es la que sigue a la directiva *ORG 0x00*. La primera instrucción ejecutada por el microcontrolador cuando se enciende o después de un RESET no es *ORG 0x00*, sino *goto INICIO* porque *ORG 0x00* es una directiva del Ensamblador, no una instrucción del microcontrolador.

En la dirección cuatro de la memoria de programa se encuentra el 'Vector de Interrupción'. Si se genera una interrupción, el microcontrolador ejecuta la instrucción que se encuentra aquí, que es *goto INTER1*.

IV.4.1 Librería para el manejo del teclado

La siguiente línea de código, *INCLUDE <TECLADO.INC>*, “pega” la librería referida en el programa. Este fichero, presenta un conjunto de subrutinas que permiten la gestión del teclado.

El teclado es de 4x4, es decir, de 16 teclas; es un teclado tipo matricial, que tiene números (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) y letras (A, B, C, D, E, F). Estos tipos de teclados, están configurados como una matriz filas-columnas con la intención de reducir el número de líneas de entrada y salida necesarias para conectarlos con el microcontrolador. En un teclado no matricial, cada tecla necesita una línea de entrada. Es decir, en el teclado matricial de 4x4 que se utiliza, entre el microcontrolador y el teclado hay 8 líneas de conexión; mientras que si se usa un teclado no matricial, entre el microcontrolador y el teclado habría 16 líneas de conexión, una por tecla. Un teclado matricial, está organizado de tal forma que cada tecla se conecta a una fila y una columna.

En la figura siguiente, se muestra las conexiones filas-columnas del teclado al puerto B del microcontrolador PIC.

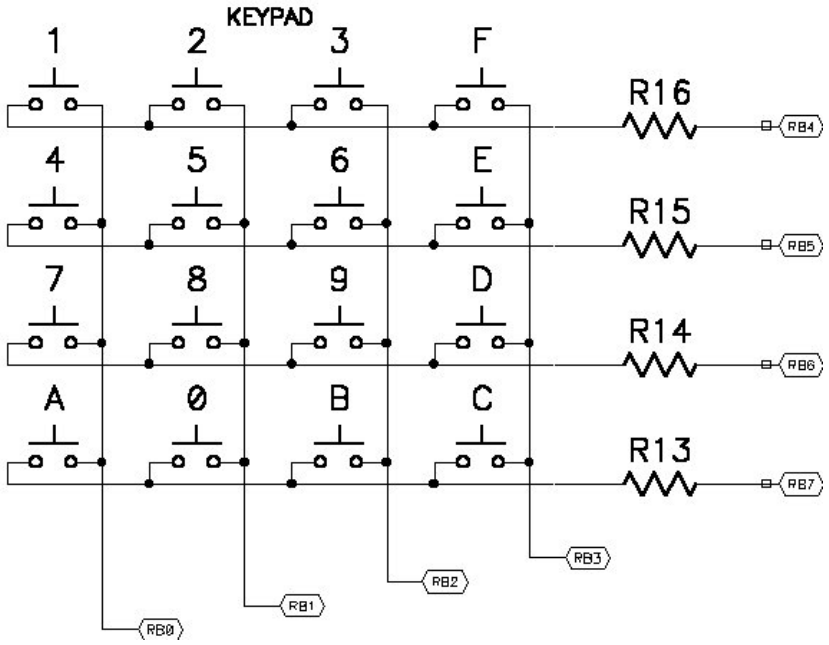

Figura 12. Teclado matricial de 4x4.

El código de las subrutinas que conforman el archivo TECLADO.INC, se encuentra en el Apéndice (páginas 78 a 80). Las subrutinas son: *Key_Scan* y *Key_BCD*.

La subrutina *Key_Scan*, realiza un barrido del teclado (explora el teclado) y detecta si hay alguna tecla pulsada. La técnica de programación que se usa para gobernar el teclado matricial se explica a continuación, basándose en el diagrama de flujo correspondiente (figura 13).

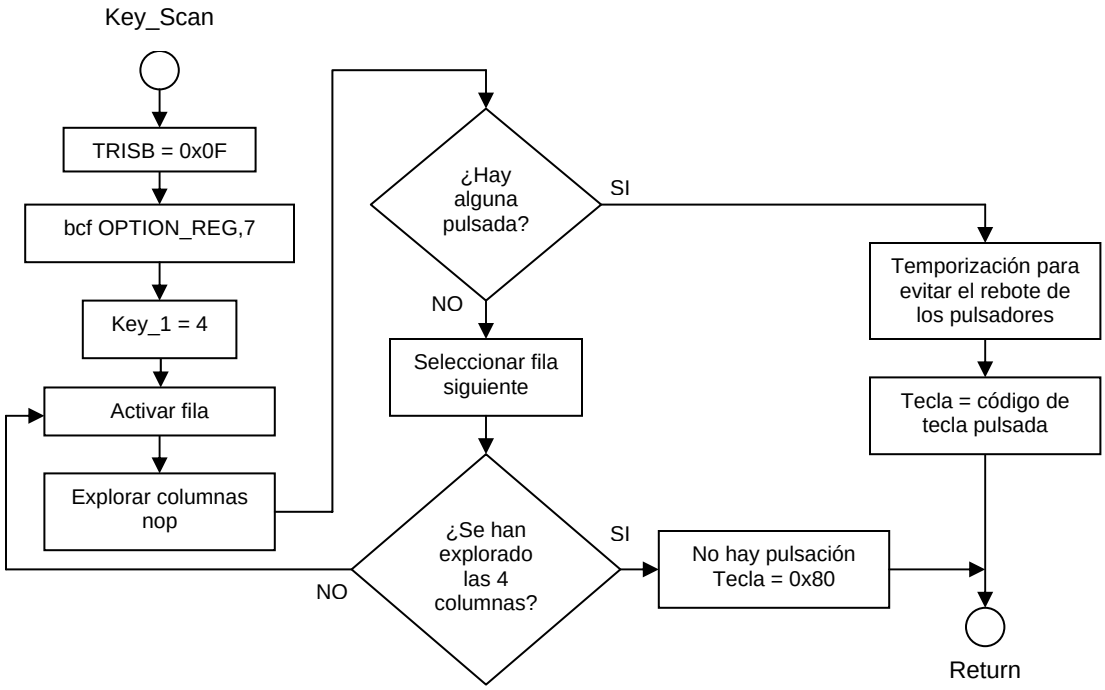

Figura 13. Diagrama de flujo de la subrutina Key_Scan.

Se configura el puerto B del microcontrolador PIC, en el cual se encuentra conectado el teclado (figura 12), para establecer la comunicación con éste. Los pines menos significativos del puerto B (RB0, RB1, RB2, RB3), se seleccionan como entradas y se conectan en ellos las columnas del teclado; por su parte las filas, se conectan a los pines más significativos (RB4, RB5, RB6, RB7) que se encuentran configurados como salidas. Esto es, el registro de control del puerto B, *TRISB*, se carga con el valor en hexadecimal de *0x0F* (figura 13).

Si no es pulsada ninguna tecla en una fila, las entradas se encuentran en estado flotante, razón por la que son necesarias resistencias de polarización, que mantienen las entradas (columnas) en un nivel alto. El PIC posee, en el puerto B, resistencias de polarización internas programables que permiten conectar el teclado matricial sin añadir componentes externos; el valor nominal de dichas resistencias es de 20 kΩ. La instrucción *bcf OPTION_REG,7* activa las resistencias de polarización programables (resistencias de pull-up).

El barrido comienza colocando a ‘0’ la primera fila y las restantes a ‘1’. Si una tecla es pulsada en la columna 0, el ‘0’ lógico aparece en la intersección fila-

columna. Las columnas son exploradas de forma secuencial comprobando si hay un ‘0’; si no se encuentra un ‘0’, se pone a ‘0’ la fila siguiente y la anterior a ‘1’ pasando a comprobar nuevamente las columnas.

La instrucción *nop* es muy importante. Siempre que el estado de un pin de salida cambia, debe transcurrir un tiempo para que se estabilice en el nuevo valor antes de que pueda ser leída. La capacidad de carga de un pin de salida, necesita un determinado tiempo para cargarse o descargarse con su nuevo valor. El intercalar una instrucción *nop*, proporciona un ciclo de retardo de tiempo suficiente para que se estabilice la señal.

La variable *Tecla* retorna al programa principal con el código de la tecla pulsada o, el *0x80* en caso de no presionarse ninguna.

Cada vez que se presiona una tecla, se producen muchos contactos intermitentes antes de que llegue al estado final. Este tipo de contactos, es lo que se denominan “rebotes”. El tiempo que dura un rebote depende del tipo de conmutador o pulsador: típicamente unos milisegundos. Si se tiene en cuenta que una instrucción se ejecuta en un tiempo comprendido de 200 ns a 20 MHz, el microcontrolador puede ejecutar fácilmente unas 5000 instrucciones por milisegundo. Al ser tan rápido, puede considerar cada rebote como el cierre de la tecla. Por lo tanto, para no considerar cada rebote como una conmutación de cierre, se introduce un retardo de tiempo por software. Este retardo, permite la “estabilización” de la tecla pulsada dando tiempo a que pase el transitorio.

Ahora, el PIC 16F877 que se utiliza tiene cinco puertos. El puerto B se ocupa para alojar el teclado, pero también se conecta en él la pantalla de cristal líquido (LCD); esto nos permite ahorrar un puerto para una futura expansión del programa. Durante la fase de lectura del teclado, la mitad del puerto B se configura como entrada y la otra mitad como salida; y, durante la escritura en el LCD, todos los pines del puerto B se configuran como salida. Entonces, con la intención de proteger y evitar que dos salidas del puerto B sean cortocircuitadas por la pulsación de una tecla, cada fila incluye una resistencia en serie de 2.2 kΩ (figura 12).

Por su parte, la subrutina *Key_BCD*, convierte el código de la tecla presionada en código BCD, del 0 al F. Antes de llamar a esta subrutina, la variable *Tecla* contiene el código de la tecla presionada; al finalizar, la subrutina devuelve el código BCD en la misma variable *Tecla*.

En esta subrutina se utiliza una tabla de datos en ROM, es decir, una lista de constantes en la memoria de programa (memoria ROM); la cual, nos permite la conversión del código de la tecla presionada a código BCD.

El diagrama de flujo de *Key_BCD* se muestra enseguida (figura 14). Hay que recordar que el código fuente de esta subrutina, se encuentra en el Apéndice (páginas 78 a 80).

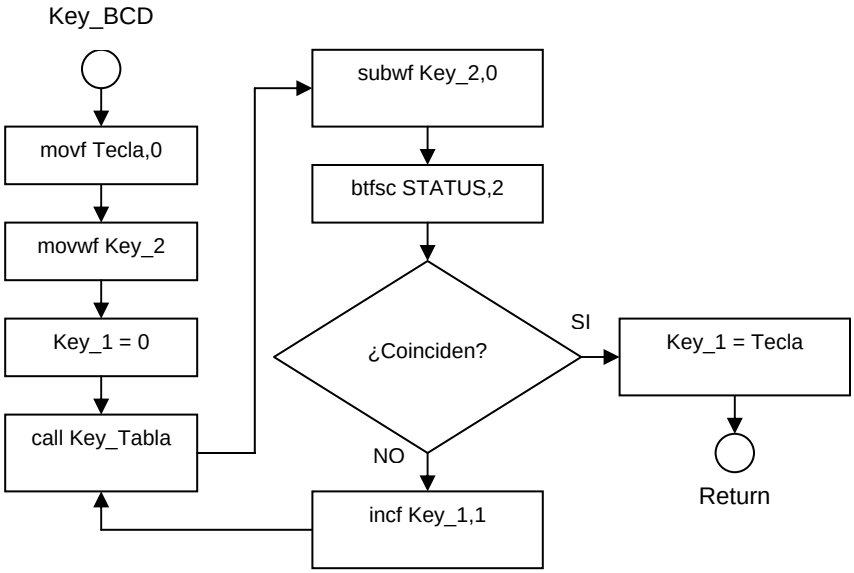

Figura 14. Diagrama de flujo de la subrutina Key_BCD.

Las dos primeras instrucciones, *movf Tecla,0* y *movwf Key_2*, hacen un respaldo del dato contenido en *Tecla* y lo almacenan en *Key_2*. Enseguida, la línea de comando *clrf Key_1* pone el valor de índice o puntero de la tabla de datos a cero. Se llama entonces a la tabla de datos (*call Key_Tabla*):

```
movf    Key_1,0
addwf   PCL,1
retlw   b'01111101'
retlw   b'11101110'
retlw   b'11101101'
⋮
```

Cada valor es leído secuencialmente, sumando el valor de índice al Contador de Programa (PC). Así, *addwf* suma el valor del puntero al byte de menos peso del PC (*PCL*) y, el resultado se almacena en *PCL*. Como es la primera vez que se ejecuta este bucle, la suma es cero y se apunta entonces a la primera instrucción *retlw*. La instrucción *retlw* funciona de manera similar a *return*: produce el retorno de una subrutina pero con un valor en el registro *W*. Entonces, *retlw b'01111101'* ejecuta el retorno del programa a la línea que sigue a la instrucción *call Key_Tabla* con el valor *b'01111101'* en *W*.

Esta línea de código (*subwf Key_2,0*), compara el código de la tecla presionada almacenado en *Key_2* con el valor leído de la tabla. Si el caracter devuelto por *retlw* no es el mismo, se incrementa el puntero de la tabla de datos (*incf Key_1,1*) y se llama nuevamente a la tabla; *addwf PCL,1* añade en este

momento un uno al *PCL* desplazando el Contador de Programa a la línea siguiente, *retlw b'11101110'*.

En caso contrario, que el caracter devuelto por *retlw* si es igual al que se encuentra en *Key_2*, el valor que tiene *Key_1* se carga en el registro *Tecla*.

IV.4.2 Librería para el manejo del LCD

Volviendo al código fuente del programa principal PROYECTO.ASM (Apéndice, página 67), la siguiente línea de código del programa después de *INCLUDE <TECLADO.INC>*, es la referente a la directiva que incorpora la librería que controla la pantalla de cristal líquido, *INCLUDE <LCD_CXX.INC>*.

Se observa que entre estas directivas y después de la que "pega" la librería para el LCD, existen varias subrutinas (entre ellas una de interrupción); éstas, se explicarán más adelante conforme sean "llamadas" por el programa principal.

La pantalla de cristal líquido (LCD) que se usa, es de la marca TIANMA Microelectronics modelo TM162AAC6-2. Se trata de un display de 2 líneas por 16 caracteres, cada uno de ellos de 5x7 pixeles y cursor (figura 15). Su consumo es muy reducido (7.5 mW) y, su fácil manejo lo hacen ideal para dispositivos que necesitan una capacidad de visualización pequeña o media, como es nuestro caso.

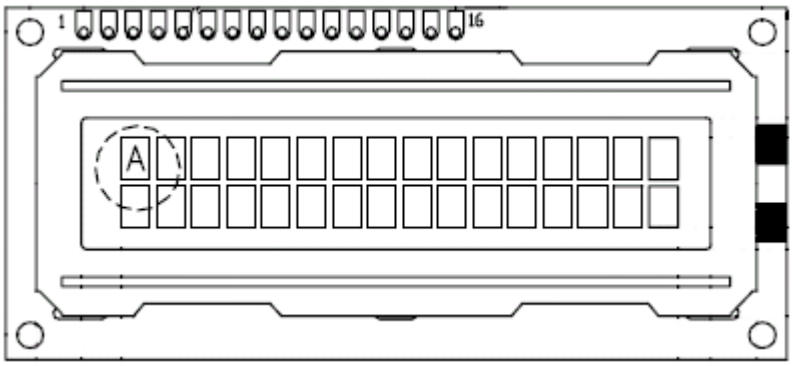

Figura 15. Pantalla de cristal líquido (LCD).

Como se ve en la figura de arriba, el módulo LCD posee 16 pines, cuya descripción se muestra en la Tabla 2. La alimentación (V_{DD}) es de +5V (típica), con un máximo de +5.5V; la regulación del contraste, se realiza mediante un voltaje obtenido al dividir V_{DD} con un potenciómetro de 10 kΩ.

Número de pin	Símbolo	Función
1	V_{SS}	Tierra
2	V_{DD}	Voltaje de alimentación
3	V_{EE}	Ajuste de contraste
4	RS	Selección de modo
5	$R/\overline{W}$	Lectura / Escritura
6	E	Validación
7	DB0	1ª línea de datos LSB
8	DB1	2ª línea de datos
9	DB2	3ª línea de datos
10	DB3	4ª línea de datos
11	DB4	5ª línea de datos
12	DB5	6ª línea de datos
13	DB6	7ª línea de datos
14	DB7	8ª línea de datos MSB
15	BGP	Background (+)
16	BGN	Background (-)

Tabla 2

La pantalla de cristal líquido (LCD), se conecta al puerto B del microcontrolador mediante un bus de 8 bits. Este bus, está formado por los pines del 7 al 14 (DB0 a DB7) del LCD. Estas líneas de datos son triestado y, pasan a estado de alta impedancia cuando el LCD no está activado, lo que desconecta a éste del PIC.

Hay tres líneas de control en el LCD. El pin 6 (línea E), de Enable o Validación, determina que el LCD, cuando está habilitado, explore el estado de las otras dos líneas de control y responda de acuerdo con el mismo. Si no está habilitado, el LCD ignora las restantes líneas y permanece inactivo.

El pin 5 (línea $R/\overline{W}$), indica si se van a leer o escribir datos en la pantalla (en su memoria RAM). El pin 4 (línea RS), determina cuándo los datos que se envían a la pantalla se están manejando como comandos o como caracteres. En la tabla siguiente, se describe el nivel de cada pin de control.

E	0 – LCD desactivado	1 – LCD activado
$R/\overline{W}$	1 – Lectura	0 – Escritura
RS	0 – Comandos	1 – Caracteres

Tabla 3

Las líneas de control de la pantalla de cristal líquido (LCD), se conectan al puerto E del microcontrolador PIC.

El código de las subrutinas que forman parte de la librería *LCD_CXX.INC*, se localiza en el Apéndice (páginas 81 a 83). Estas subrutinas son: *UP_LCD*, *LCD_BUSY*, *LCD_E*, *LCD_DATO*, *LCD_REG*, *LCD_INI*, *LCD_DELAY*, *CHECARE*, *CHECARD*, *CHECARB* y *CHECARA*.

La subrutina *UP_LCD*, configura los puertos del microcontrolador PIC que se usan para la comunicación con la pantalla de cristal líquido (LCD), los puertos B y E.

La subrutina *LCD_BUSY*, por su parte, es utilizada por otras subrutinas dentro de la misma librería *LCD_CXX.INC*. El controlador del LCD, tarda como máximo 40 μs en completar una lectura o una escritura. Para evitar hacer esperar al PIC y escribir en la pantalla de cristal líquido cuando hay una operación interna en curso, el LCD cuenta con una bandera de ocupado BF (Busy Flag). Cuando dicha bandera está a '1', no se puede leer ni escribir en el LCD, está ocupado; cuando BF = 0, la pantalla de cristal líquido se encuentra libre. Entonces, esta subrutina (*LCD_BUSY*) se encarga precisamente, de checar la bandera de ocupado del LCD, e indicarnos cuándo podemos escribir en él.

LCD_E, también es una subrutina empleada por otras dentro de *LCD_CXX.INC*. Se encarga de enviar pulsos de habilitación al pin 6 (línea E) del LCD. Cuando dicha línea de control pasa a un nivel bajo, la pantalla de cristal líquido es deshabilitada y todas sus líneas de E/S pasan a un estado de alta impedancia, lo que desconecta al LCD del PIC. Debe colocarse a un nivel alto, cuando se desea escribir o leer comandos o caracteres en o de la pantalla. La duración del pulso necesario debe superar los 500 ns, según las hojas de especificaciones del LCD.

La línea de control RS (pin 4 del LCD), como se mencionó, selecciona entre modo comando (RS=0) o modo caracter (RS=1). Las subrutinas *LCD_DATO* y *LCD_REG*, tratan este pin; *LCD_REG* envía comandos a la pantalla de cristal líquido, mientras que *LCD_DATO*, se usa para escribir caracteres en la memoria RAM del LCD.

Ahora, antes de utilizar la pantalla LCD, es necesario inicializarla. La subrutina encargada de dicha acción es *LCD_INI*. El diagrama de flujo correspondiente se muestra en la figura 16.

Esta subrutina, comienza enviando el comando 'Function Set' 3 veces seguidas con un intervalo de aproximadamente 5 ms.

Este comando, configura a la pantalla de cristal líquido con los siguientes parámetros: 8 bits de datos, dos líneas de pantalla y, 16 caracteres con una matriz de 5x7.

El retardo de 5 ms después de enviar el comando, se logra con la subrutina *LCD_DELAY*, la cual, se trata de un bucle finito anidado.

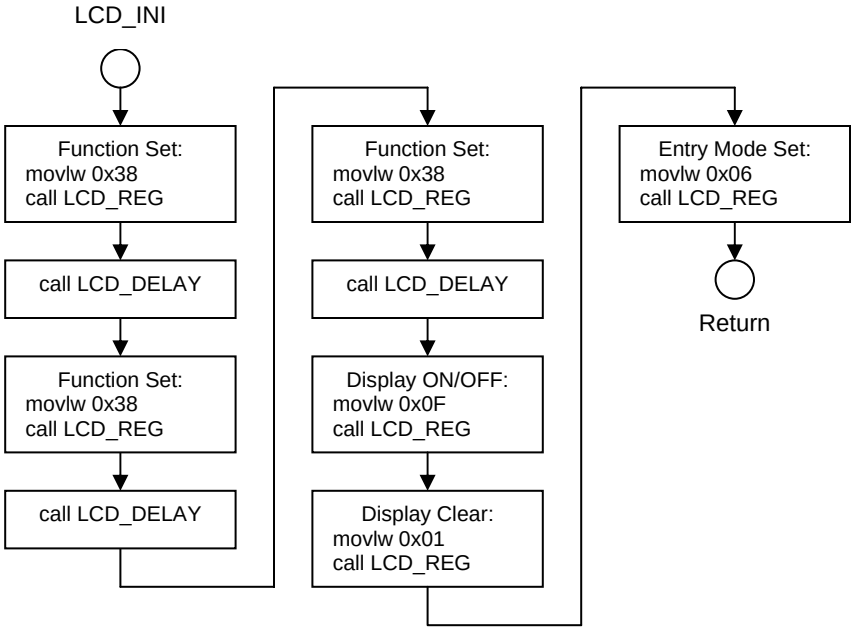

Figura 16. Diagrama de flujo de la subrutina LCD_INI.

Después del tercer retardo, se envía a la pantalla de cristal líquido el comando 'Display ON/OFF' con un valor de *0x0F* (*b'00001111'*); esta instrucción, enciende la pantalla, el cursor y, hace parpadear a este último y que adquiera la forma de ▉.

Enseguida, se envía la instrucción 'Display Clear' (*b'00000001'*), que borra toda el área del display y, devuelve el cursor a la posición 1 línea 1 (posición inicial). Finalmente, se manda el comando 'Entry Mode Set', que configura al LCD para que incremente la posición del cursor a la derecha después de cada caracter.

Es importante mencionar que, el orden en que se ejecutan las líneas de código que conforman la subrutina *LCD_INI*, debe respetarse, ya que si no es así, la pantalla de cristal líquido no funcionará.

Por otra parte, cada línea de la pantalla LCD tiene 40 caracteres de RAM de pantalla de datos (DDRAM) asociados. Las 40 posiciones de RAM, están organizadas con un buffer de forma circular de tal forma que la última posición enlaza con la primera. En el caso del LCD utilizado, sólo son visibles 16 de estos 40 caracteres en la ventana de la pantalla (figura 17).

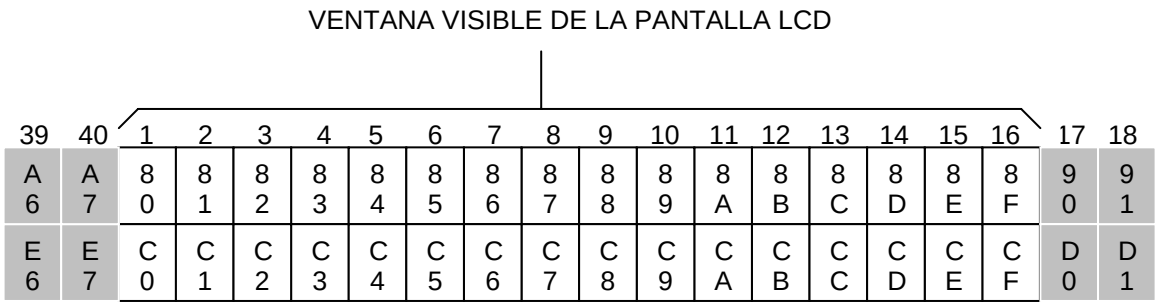

Figura 17. Posición de los caracteres en la dirección DDRAM.

Las subrutinas *CHECARE*, *CHECARD*, *CHECARB* y *CHECARA*, colocan el cursor en una posición específica de la pantalla de cristal líquido para escribir en ella un caracter relacionado con el tiempo establecido.

Por ejemplo, *CHECARB*, coloca el cursor en la posición *0xCB* de la ventana visible del LCD para escribir en ella la unidad de minuto del tiempo establecido.

IV.4.3 Programa principal

Hasta aquí, se ha explicado la cabecera del código fuente del programa para el sistema de control de la temperatura de la termoformadora (PROYECTO.ASM); así como, las librerías referentes al manejo del teclado y de la pantalla de cristal líquido (LCD).

Ahora, recordando, la directiva *ORG* le indica al lenguaje Ensamblador dónde debe comenzar a colocar las instrucciones en la memoria de programa (memoria ROM); la dirección de inicio es en la posición cero.

Retomando el código fuente del programa principal (PROYECTO.ASM) en la página 67 del Apéndice, se observa que la instrucción *goto INICIO* ha sido colocada en la dirección cero de la memoria ROM, que es la que sigue a la directiva *ORG 0x00*. Así, la primera instrucción ejecutada por el microcontrolador PIC cuando se enciende o después de un RESET, es *goto INICIO*.

Esta instrucción, provoca que el microcontrolador salte a la dirección apuntada por la etiqueta *INICIO* (página 71 del Apéndice). En esta sección del código fuente, comienza propiamente el programa.

A dicha sección, se le llama ‘programa principal’, y su diagrama de flujo se muestra en la figura 18.

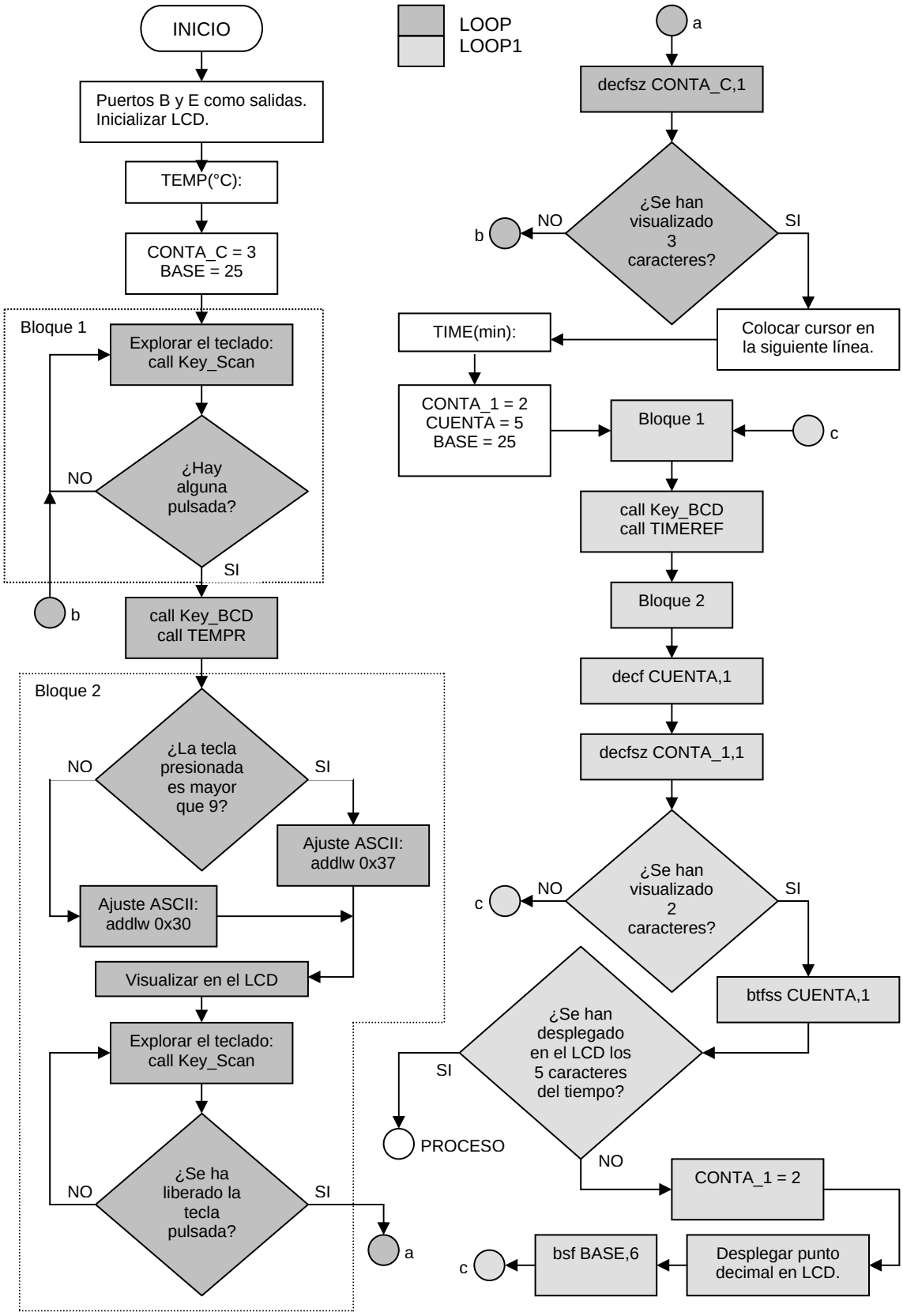

Figura 18. Diagrama de flujo del programa principal.

Este segmento del programa, 'programa principal', nos permite el ingreso de la temperatura de operación deseada del horno de la máquina de termoformado y, el tiempo que dicho horno permanecerá encendido a tal temperatura.

Primero, se configuran los puertos B y E como salidas. Hecho lo cual, se inicializa el LCD (subrutinas *UP_LCD* y *LCD_INI*). Entonces, la pantalla de cristal líquido está lista para desplegar el primer mensaje del programa: *TEMP(°C):* ; el cual, nos indica que ingresemos la temperatura de operación deseada del horno de la máquina de termoformado (en el rango de 0 a 240 °C).

Después de visualizar el mensaje *TEMP(°C):* , se inicializan los registros *CONTA_C* y *BASE*. *CONTA_C* es el contador de caracteres, y se carga con el número 3 en decimal; esta constante, representa el número de caracteres de que consta la temperatura deseada del horno así como también, el número de veces que se ejecutará el bucle siguiente (*LOOP*). *BASE* por su parte, se carga con el número 25 en decimal (*b'00011001'*); este número, permite almacenar la temperatura deseada o de referencia.

Enseguida, se llega al bucle *LOOP*. Un bucle, permite básicamente la repetición del código del programa y, puede funcionar de forma indefinida o repetir una parte del código un número determinado de veces (como es nuestro caso). Con el bucle *LOOP*, se ingresa la temperatura deseada o de referencia del horno de la máquina de termoformado, la cual consta de tres caracteres para cualquier valor; es decir, para una temperatura de 40 °C por ejemplo, se debe de teclear 040.

La primera instrucción del bucle *LOOP* es, *call Key_Scan*; la cual, llama a la subrutina *Key_Scan* para explorar el teclado y detectar si se presionó una tecla. Si no se presiona ninguna tecla, el programa continúa "barriendo" el teclado; caso contrario, se llama entonces a la subrutina *Key_BCD*, la cual, convierte el código de la tecla presionada en código BCD, del 0 al F. Antes de llamar a esta subrutina, la variable *Tecla* contiene el código de la tecla presionada; al finalizar, la subrutina devuelve el código BCD en la misma variable *Tecla*.

La siguiente línea de código es *call TEMPR*, la cual llama a la subrutina *TEMPR*, cuyo código fuente se encuentra en la página 68 del Apéndice. Esta subrutina, almacena cada tecla pulsada de la temperatura de referencia en los registros *TRBCD0*, *TRBCD1* y *TRBCD2*, en código BCD. El registro *TRBCD0*, guarda el dígito más significativo de la temperatura deseada; en el *TRBCD1*, se guarda el dígito intermedio, y en el registro *TRBCD2*, se almacena el dígito menos significativo de la temperatura de referencia. La constante contenida en el registro *BASE* (25 en decimal), es desplazada un bit a la derecha cada vez que se ejecuta la subrutina *TEMPR* (se ejecuta tres veces durante el bucle *LOOP*), permitiendo

almacenar los tres caracteres que conforman la temperatura deseada del horno de la máquina termoformadora.

Una vez almacenado el código BCD de la tecla presionada, es necesario desplegar tal caracter en la pantalla de cristal líquido (LCD), para que podamos comprobar que efectivamente es el valor deseado; para ello, se ajusta el código BCD. El LCD, posee una pequeña memoria ROM donde se tiene "de fábrica" caracteres ASCII, japoneses, griegos y símbolos matemáticos; a esta memoria, se le llama generador de caracteres CGRAM (Character Generator RAM). Entonces, se verifica si la tecla presionada es mayor a 9 o no, para así, saber qué tipo de ajuste se va a realizar.

Si no es mayor que 9, al código BCD se le suma 30 en hexadecimal (*0x30*), para llamar así, al caracter determinado del CGRAM (específicamente, del 0 al 9). Si la tecla presionada si es mayor a 9, al código BCD se le suma 37 en hexadecimal (*0x37*), llamando así, al caracter correspondiente del CGRAM (de la A a la F).

Una vez ajustado el código BCD de la tecla presionada, se llama a la subrutina *LCD_DATO* (*call LCD_DATO*), para visualizar el caracter.

Enseguida, se explora nuevamente el teclado para saber si se ha liberado la tecla pulsada, tras lo cual, se comprueba si se han visualizado los tres caracteres que forman a la temperatura de referencia.

Si no se han desplegado los tres caracteres de la temperatura deseada, se ejecuta una vez más el bucle *LOOP*; caso contrario, si en el LCD ya se visualizan los tres caracteres y por ende, ya se tiene almacenado la temperatura de operación deseada, se coloca el cursor del LCD en la segunda línea.

La pantalla de cristal líquido está lista en este momento, para desplegar el segundo mensaje del programa: *TIME(min):* ; éste, nos indica que ingresemos el tiempo establecido, es decir, el tiempo que el horno de la máquina de termoformado permanecerá encendido a la temperatura requerida (en el intervalo de 0 a 10 minutos).

Desplegado el mensaje *TIME(min):* , se inicializan los registros *CONTA_1*, *CUENTA* y *BASE*. *CONTA_1* es un contador de caracteres, y se carga con el número 2 en decimal; esta constante, permite primero el ingreso de los minutos del tiempo establecido y después, el de los segundos. *CUENTA*, se carga con el número 5 en decimal, que representa el número de caracteres de que consta el tiempo establecido; así también este registro, permite ejecutar el bucle siguiente (*LOOP1*), el número de veces necesario (cuatro veces) para ingresar el tiempo establecido. *BASE* por su parte, también se utiliza para almacenar el tiempo establecido; se carga nuevamente con el número 25 en decimal (*b'00011001'*).

Se llega así, al bucle *LOOP1*. Este bucle, permite ingresar el tiempo que el horno de la máquina de termoformado permanecerá encendido a la temperatura deseada, y cuyo formato para cualquier valor es el siguiente: ##.## . Es decir, para un tiempo establecido de 9 minutos 25 segundos por ejemplo, primero se teclea 09 y enseguida 25. Los minutos pueden ser 00, 01, 02... hasta 10; los segundos en tanto, pueden ser 00, 01, 02... hasta 59.

La primera instrucción de *LOOP1* es, *call Key_Scan*; la cual, llama a la subrutina *Key_Scan* para explorar el teclado y detectar si se presionó una tecla. Si no se presiona ninguna tecla, el programa continúa "barriendo" el teclado; caso contrario, se llama entonces a la subrutina *Key_BCD*, la cual, convierte el código de la tecla presionada en código BCD, del 0 al F. Antes de llamar a esta subrutina, la variable *Tecla* contiene el código de la tecla presionada; al finalizar, la subrutina devuelve el código BCD en la misma variable *Tecla*.

Se ejecuta entonces *call TIMEREF*, el cual llama a la subrutina *TIMEREF*, cuyo código fuente se encuentra en la página 68 del Apéndice. Esta subrutina, almacena cada tecla pulsada del tiempo establecido en los registros *ACLOCK*, *BCLOCK*, *DCLOCK* y *ECLOCK*, en código BCD. El registro *ACLOCK*, guarda el dígito más significativo, la decena de minuto del tiempo establecido; en el *BCLOCK*, se guarda la unidad de minuto del tiempo establecido; en *DCLOCK*, se aloja la décima de segundo del tiempo establecido; y en el registro *ECLOCK*, se almacena el dígito menos significativo, la centésima de segundo del tiempo establecido. La constante contenida en el registro *BASE* (25 en decimal), es desplazada un bit a la derecha cada vez que se ejecuta la subrutina *TIMEREF* (se ejecuta cuatro veces durante el bucle *LOOP1*), permitiendo almacenar los dos caracteres que conforman los minutos del tiempo establecido y, los dos caracteres correspondientes a los segundos del tiempo establecido.

Una vez almacenado el código BCD de la tecla presionada, se verifica si ésta es mayor a 9 o no, para así, saber qué tipo de ajuste se va a realizar.

Si la tecla no es mayor que 9, al código BCD se le suma 30 en hexadecimal (*addlw 0x30*). Si la tecla presionada si es mayor al número 9, al código BCD se le suma 37 en hexadecimal (*addlw 0x37*).

Entonces, ya ajustado el código BCD de la tecla presionada, se despliega tal caracter en la pantalla de cristal líquido (LCD), para comprobar que efectivamente es el valor deseado.

Enseguida, se explora nuevamente el teclado para saber si se ha liberado la tecla pulsada, tras lo cual, se comprueba si se han visualizado los dos caracteres que forman los minutos del tiempo establecido (*decfsz CONTA_1,1*). Si la respuesta es no, se ejecuta una vez más el bucle *LOOP1*. Si la respuesta es si, se comprueba ahora si se han desplegado ya los 5 caracteres que conforman al tiempo establecido (*btfss CUENTA,1*).

Si en el LCD, no se observa los 5 caracteres que constituyen el tiempo establecido, el registro *CONTA_1* se vuelve a cargar con el número 2 en decimal, se despliega el punto decimal en la pantalla de cristal líquido y, se salta una vez más al inicio del bucle *LOOP1* para poder ingresar ahora los segundos del tiempo establecido.

En caso contrario, que ya aparezcan en la pantalla del LCD los 5 caracteres del tiempo establecido y por consecuencia éste se encuentre almacenado, se ejecuta la línea de código *goto PROCESO*. Esta instrucción, permite salir del bucle *LOOP1* y de la sección de código 'programa principal', e ir a la parte del programa PROYECTO.ASM llamada 'PROCESO1' (página 73 del Apéndice).

IV.4.4 Proceso1

Este segmento de PROYECTO.ASM, realiza la conversión de la señal analógica proveniente de la etapa del sensor-acondicionador, en una señal digital; visualiza en el LCD el valor de la temperatura sensada y, compara dicho dato con la temperatura de referencia. En la figura 19, se presenta el diagrama de flujo correspondiente.

El inicio de esta sección, se encuentra indicada por la etiqueta *PROCESO*. En primer lugar, se configura el tipo de interrupción a utilizar. La interrupción, es una técnica que coloca al programa temporalmente en suspenso mientras el microcontrolador ejecuta otro conjunto de instrucciones en respuesta a un suceso. Las causas de una interrupción pueden ser externas, como la activación de un pin con el nivel lógico apropiado, e internas, como la que puede producirse al finalizar una conversión A/D.

El modelo de PIC que se usa, el PIC16F877, tiene 14 posibles causas que pueden originar una interrupción. En nuestro caso, se emplea la interrupción por desbordamiento del Timer1 (TMR1); por lo que se programan adecuadamente los bits de los registros relacionados, *PIE1* e *INTCON*, para habilitarla.

El TMR1, es un Temporizador/Contador ascendente con un tamaño de 16 bits, lo que requiere el uso de dos registros concatenados de 8 bits: *TMR1H*:*TMR1L*, que son los encargados de guardar el valor del contaje en cada momento. El valor almacenado es *0x3CB0*, a partir del cual, el TMR1 empieza la temporización para lograr una interrupción cada 10 ms. Antes de ingresar el valor a partir del cual el TMR1 empieza el conteo, éste se desactiva.

El registro *RELOJ* por su parte, se inicializa con un valor en hexadecimal de *0x63*. Este valor, permite al programa desplegar en el LCD, el tiempo que resta del tiempo establecido en intervalos de 1 s. Este despliegue, se realiza aprovechando la rutina de servicio de interrupción, como más adelante se explica al detallar dicha rutina.

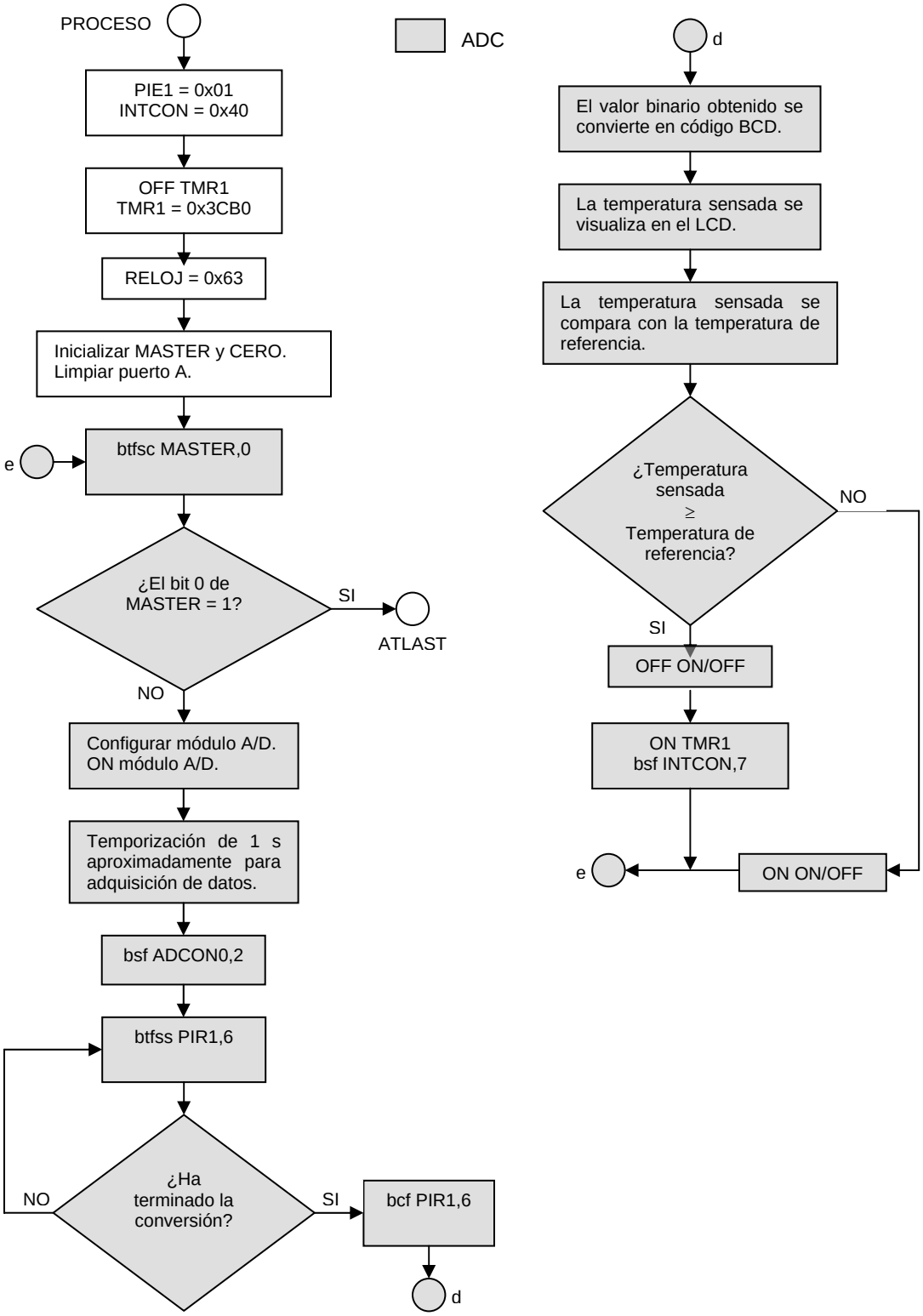

Figura 19. Diagrama de flujo de PROCESO1.

Enseguida, se ponen a cero los registros *MASTER* y *CERO*; así también, se limpia el puerto A. *MASTER*, nos permite salir de 'PROCESO1' y pasar a la etapa de la electroválvula, una vez terminado el tiempo establecido. *CERO* en tanto, permite saber, en la rutina de tratamiento de interrupción, si alguno de los caracteres que conforman al tiempo establecido es igual a cero.

Se llega así, al bucle *ADC*. La primera línea de código, comprueba si el bit 0 de *MASTER* es igual a '1' (*btfsc MASTER,0*); si la respuesta es sí, el programa abandona la sección 'PROCESO1' y, salta al segmento de PROYECTO.ASM que se refiere a la etapa de la electroválvula (*ATLAST*). Si el resultado es no, se continua con el bucle *ADC*.

Entonces, se configura el módulo conversor analógico/digital. Los microcontroladores PIC16F877, poseen un conversor A/D de 10 bits de resolución y 8 canales. La resolución que tiene cada bit procedente de la conversión, tiene un valor que es función de la tensión de referencia (Vref), de acuerdo con la fórmula siguiente:

$$\text{Resolución} = \frac{(+Vref) - (-Vref)}{1024} = \frac{Vref}{1024}$$

Así, en nuestro caso, el +Vref = 5.12 VDC (que es además, la tensión de alimentación V_{DD} del microcontrolador) y la –Vref es tierra (V_{SS}). La razón de estas magnitudes se debe a que con ellas, se logra una resolución de 5 mV/bit, lo que simplifica las operaciones de conversión de binario a código BCD (detalladas un poco más adelante). Por lo tanto, a la entrada analógica de 0 V, le corresponde una digital de '00 0000 0000' y, para la de 5.12 V, un valor en binario de '11 1111 1111'. La señal analógica, se obtiene de la etapa del sensor-acondicionador, y posee una tasa de 10 mV/°C.

El módulo A/D, se programa con las siguientes características: se utiliza el pin 0 del puerto A (RA0), como canal de entrada analógica; justificación a la derecha, esto es, en la pareja de registros *ADRESH*:*ADRESL*, se deposita el resultado de la conversión, que al estar compuesto por 10 bits, sólo son significativos los 10 bits de menor peso de dicha pareja, mientras que los 6 de más peso, son iguales a cero. Así también, el tiempo que dura la conversión de cada bit (T_{AD}), se selecciona con un valor de 1.6 μs, tiempo recomendado para la frecuencia de funcionamiento de este tipo de microcontrolador (20 MHz).

Se activa el conversor A/D, y entonces, se da un tiempo de espera de aproximadamente 1 s para la adquisición de datos por parte del módulo. Una vez transcurrida la temporización, se da inicio a la conversión (*bsf ADCON0,2*). Al terminar ésta, se restaura el 'flag' del conversor (*bcf PIR1,6*), para una nueva conversión A/D.

En este momento, la pareja de registros *ADRESH*:*ADRESL*, contiene el resultado digital de la conversión de la señal analógica correspondiente, de 10 bits de longitud. *ADRESH*, almacena la parte alta del resultado de la conversión A/D; mientras que *ADRESL*, guarda la parte baja.

El valor que se obtiene del módulo A/D representa, en el sistema numérico binario, la temperatura del horno de la máquina de termoformado en un momento dado, la temperatura sensada. Para desplegar tal dato en el LCD y compararlo con la temperatura de referencia, es necesario convertirlo de binario a código BCD. En la parte superior derecha del diagrama de flujo de la figura 19, se observa el bloque relacionado con esta operación. En la figura siguiente, se detalla dicho bloque.

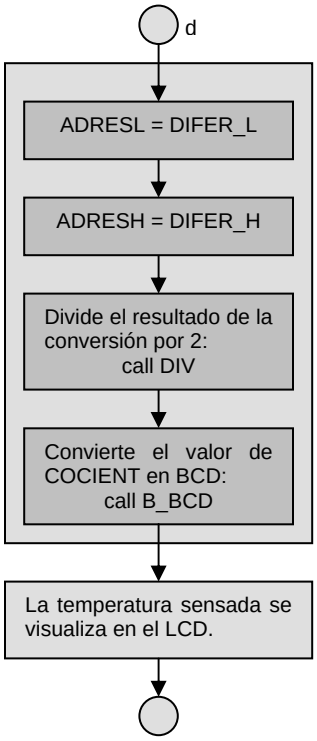

Figura 20. Diagrama de flujo del bloque de conversión Binario-BCD.

Antes de realizar las operaciones que se requieren para pasar de binario a código BCD, la parte baja y alta del resultado de la conversión A/D se cargan en los registros *DIFER_L* y *DIFER_H*, respectivamente.

Ahora, el dato binario que resulta del módulo A/D, es necesario convertirlo a un número binario cuyo valor en decimal sea el de la temperatura sensada. Para ello, primero se multiplica el dato del conversor por su resolución (5 mV/bit) y enseguida, al resultado que se obtiene de esta operación se divide por la tasa de la etapa del sensor-acondicionador (10 mV/°C).

Por ejemplo, supongamos un dato binario de b'00 0011 0110' proveniente del conversor A/D; su equivalente en decimal es 54. Entonces, la temperatura sensada que representa es la siguiente:

$$\text{Temperatura sensada} = (54)(5mV/bit) = 270mV$$

$$\text{Temperatura sensada} = \frac{270mV}{10mV/°C} = 27°C$$

El resultado es 27 °C, que representado en el sistema numérico binario es igual a, b'00 0001 1011'.

Si comparamos el binario antes de las operaciones y el que se tiene después, se observa que este último es el mismo que el valor inicial con un desplazamiento (corrimiento) a la derecha un lugar. Esto, equivale a la división por 2.

De hecho, simplificando la resolución del conversor con la tasa de la señal analógica en las operaciones, basta con dividir el resultado del conversor (en el ejemplo de arriba, 54 en decimal) por 2, para obtener el binario de la temperatura sensada.

Así entonces, para la implementación por software de estas operaciones, se utiliza la división del resultado del módulo A/D por 2 (subrutina *DIV*, página 68 del Apéndice); que no es más que un corrimiento a la derecha un lugar, como se acaba de ver. El resultado de esta división, de este corrimiento, se almacena en el registro *COCIENT*.

Es importante aclarar, que todos los datos que se manejan son enteros. La razón de ello, es que facilitan en gran medida las operaciones y su implementación en el programa; además, la aplicación nos permite "perder" decimales, sin afectar sensiblemente las lecturas de temperatura.

Ya con el número binario (cuyo valor en decimal es el de la temperatura sensada) en *COCIENT*, se llama a la subrutina *B_BCD* (página 69 del Apéndice); la cual, convierte dicho valor a código BCD. El resultado, se deposita en los registros *CODIGOH* y *CODIGOL*. En *CODIGOH*, se guarda la parte alta del resultado de la conversión de binario a BCD; y en *CODIGOL*, la parte baja.

Hay que mencionar, que la subrutina *B_BCD*, sólo pasa de binario a BCD datos de hasta un máximo de 8 bits de longitud. Esto significa que, el rango de temperatura que puede manejar el programa PROYECTO.ASM, es de 0 a 255 °C.

Una vez con el dato de la temperatura sensada en código BCD, ésta se visualiza en la pantalla de cristal líquido LCD. Para lo cual, es necesario separar antes el valor almacenado en *CODIGOL* en dos valores. Un valor, se obtiene de la siguiente forma: se copia *CODIGOL* en el registro *LAPSO* y, los cuatro bits más

significativos de este último se ponen a cero. Entonces en *LAPSO*, se tiene la unidad de la temperatura sensada. El segundo valor por su parte, se forma poniendo a cero los 4 bits menos significativos de *CODIGOL* e, intercambiándolos con los 4 más significativos. Se tiene así, la decena de la temperatura del sensor.

La separación de *CODIGOL* en dos valores, se realiza no sólo para poder desplegar la temperatura sensada en el LCD, sino también, para poder compararla con la temperatura de referencia y así, generar la señal de control ON/OFF. *CODIGOH*, contiene la centena de la temperatura sensada.

Retomando el diagrama de flujo de la figura 19, después de observar en el LCD la temperatura del sensor, el siguiente paso es hacer la comparación entre ésta y la temperatura deseada del horno de la máquina termoformadora. En la figura que se muestra a continuación, se presenta con mayor detalle esta parte.

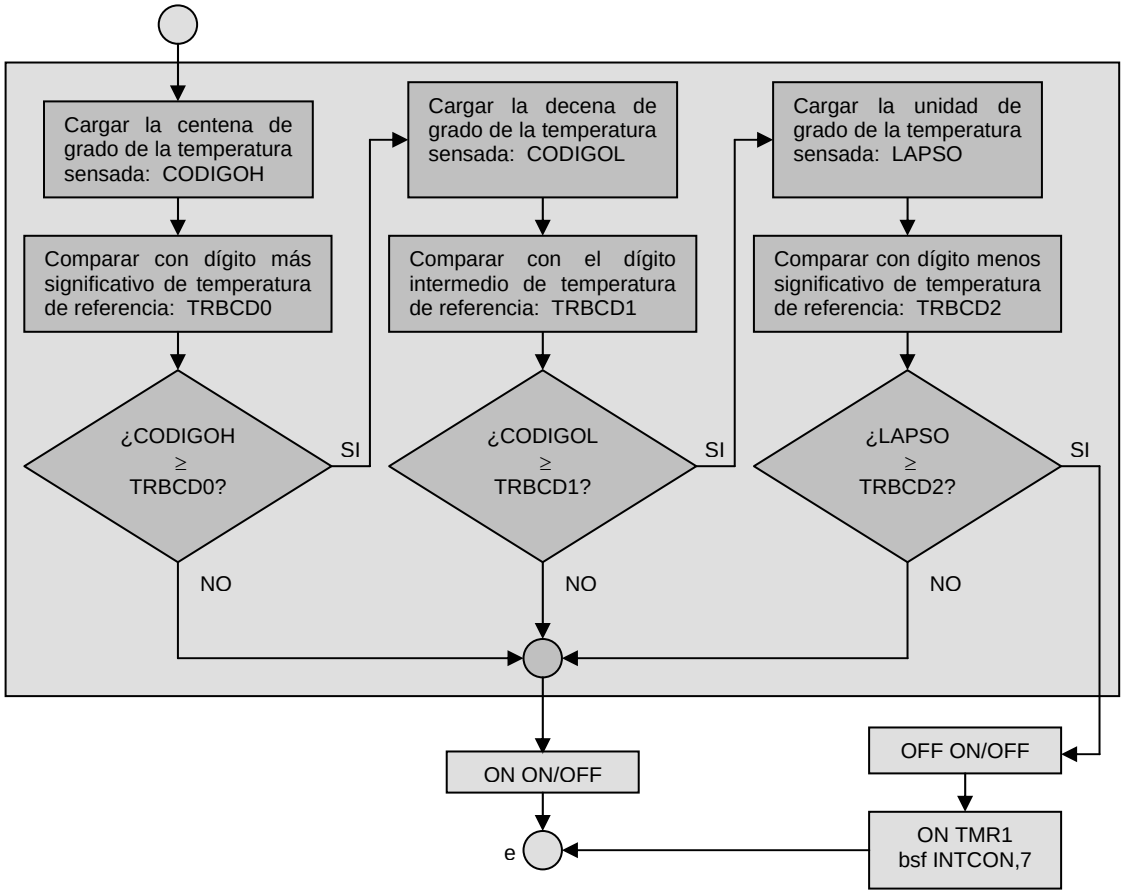

Figura 21. Diagrama de flujo de la comparación de temperaturas.

Primero, se compara la centena de la temperatura sensada almacenada en el registro *CODIGOH*, con el dígito más significativo de la temperatura deseada que se encuentra en *TRBCD0*. Si son iguales o, el caracter de la temperatura del

sensor es mayor, se compara entonces la decena de grado de ésta (*CODIGOL*), con el dígito intermedio de la temperatura de referencia (*TRBCD1*).

Si resulta que otra vez son iguales o, que el valor de la temperatura sensada es mayor que el correspondiente a la temperatura de referencia, se compara ahora la unidad de grado de la primera (*LAPSO*), con el dígito menos significativo de la última (*TRBCD2*).

Si una vez más son iguales o, *LAPSO* es mayor que *TRBCD2*, se apaga la señal de ON/OFF; con lo cual, se apaga la alimentación del horno de la termoformadora. Enseguida, se activa el Timer1 (TMR1) y, el permiso global de interrupciones. Entonces, se vuelve al inicio del bucle *ADC* para una nueva conversión analógico-digital (*goto ADC*).

Ahora, si en las comparaciones anteriores, cualquiera de los caracteres que conforman a la temperatura sensada es menor que sus correspondientes de la temperatura de referencia, se enciende la señal de ON/OFF. Lo que sigue, es una nueva conversión A/D (*goto ADC*).

IV.4.5 Rutina de tratamiento de interrupción INTER1

Recopilando lo explicado hasta aquí del código fuente del programa para el sistema de control de la temperatura de la termoformadora PROYECTO.ASM, tenemos que:

Al iniciar el sistema de control para un proceso de termoformado, en primer lugar, se despliega en la pantalla de cristal líquido (LCD) el mensaje *TEMP(°C):* ; se ingresa entonces (vía teclado), la temperatura de operación deseada del horno de la máquina de termoformado en el rango de 0 a 240 °C, la cual, también se visualiza en la pantalla LCD. Esta entrada, representa la temperatura de referencia.

Enseguida, aparece en el LCD el mensaje *TIME(min):* ; con el teclado se indica entonces, el tiempo que se mantendrá encendido el horno a la temperatura requerida, en el intervalo de 0 a 10 minutos; dicho dato, también se observa a través del LCD. Esta entrada, representa el tiempo establecido.

La sección de código ‘programa principal’, trata lo anterior. Si en algún momento al ingresar la temperatura de referencia y/o el tiempo establecido, nos equivocamos, hay que reiniciar el proceso de termoformado presionando la tecla de RESET.

Con la temperatura de referencia y el tiempo establecido ya especificados, se pasa al segmento ‘PROCESO1’. Esta parte de PROYECTO.ASM, cuenta con el bucle *ADC*, el cual, lleva a cabo la secuencia siguiente: primero, la señal analógica que proviene de la etapa del sensor-acondicionador (temperatura

sensada), se convierte en una señal digital de 10 bits. Este resultado, que se encuentra representado en el sistema numérico binario, se convierte a código BCD. Ya en este formato, se despliega en la pantalla LCD. La temperatura sensada es, la temperatura actual del horno de la termoformadora.

Lo que sigue es, comparar la temperatura del sensor con la temperatura de referencia. Cuando la primera temperatura es menor que la segunda, se activa la señal de control ON/OFF. Esta señal, a través de la etapa de potencia (capítulo V), enciende la máquina termoformadora. La acción, provoca que la temperatura del horno de ésta aumente gradualmente.

Por el contrario, si la temperatura sensada es igual o mayor que la de referencia, se desactiva la señal ON/OFF y por ende, se apaga la alimentación del horno de la termoformadora.

En ambos casos, que la temperatura sensada sea menor que la temperatura que se requiere o, que sea igual o mayor, se regresa al inicio de *ADC* para ejecutar una vez más su secuencia.

Este ciclo, se realiza una y otra vez; sin embargo, en algún momento debe de terminar. El encargado de ello, es el registro *MASTER*, cuando su bit 0 es igual a '1'. El bucle *ADC*, comienza comprobando este bit: si no es igual a '1', se continua con la secuencia pero, si sí es igual, se abandona el ciclo *ADC*, así como la sección 'PROCESO1' y, se salta a la etapa de la electroválvula (*ATLAST*).

El bit 0 de *MASTER*, es igual a '1' una vez que el tiempo establecido ha llegado a su fin. Este tiempo, es tratado por la rutina de servicio de interrupción *INTER1*.

Como se mencionó anteriormente, la interrupción que emplea el programa es por desbordamiento del TMR1. El diagrama de flujo correspondiente a esta interrupción (*INTER1*), se presenta en la figura 22. Su código fuente, se localiza en el Apéndice, páginas 69 a 71.

La primera vez que son iguales las temperaturas o, que la temperatura sensada es mayor que la de referencia, no sólo se apaga la señal de control ON/OFF sino también, se activa el Timer1 (TMR1) y el permiso global de interrupciones.

En ese momento, el TMR1 empieza un conteo ascendente a partir del valor *0x3CB0*, con cada ciclo de instrucción. Cuando se produce el desbordamiento del contaje, es decir, cuando se pasa de 0xFFFF a 0x0000, se activa la bandera (flag) de interrupción y se genera ésta. Esto sucede, cada 10 ms.

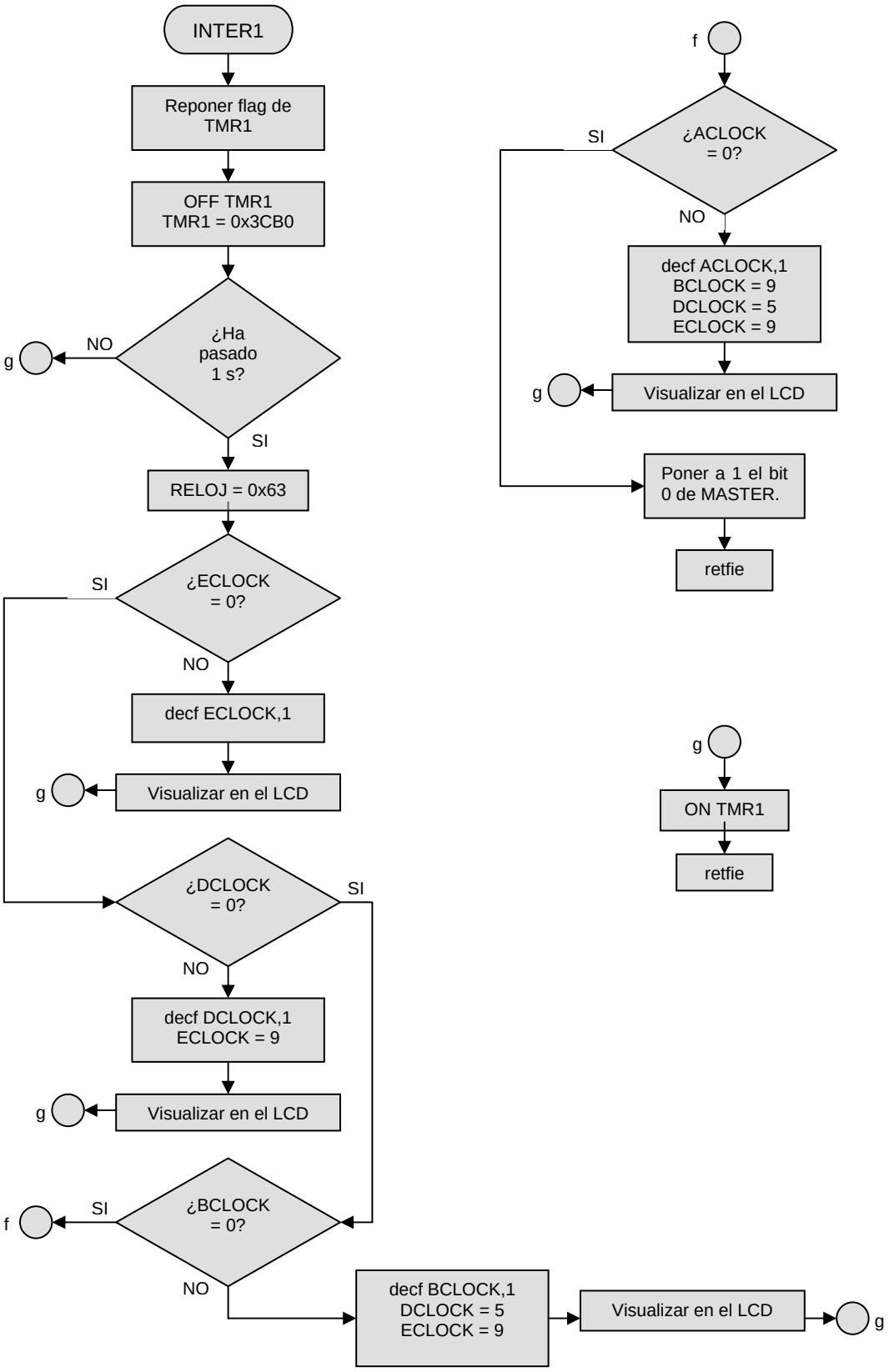

Figura 22. Diagrama de flujo de INTER1.

INTER1, comienza reponiendo el flag correspondiente; el TMR1 se desactiva y, se carga con el valor *0x3CB0* para la siguiente interrupción.

El registro *RELOJ*, tiene almacenado la constante *0x63*. Cada vez que se ejecuta una interrupción, dicha constante es decrementada en uno. Cuando se genera la interrupción número 100, *RELOJ* tiene un valor de 0x00, lo que significa que ha pasado 1 s. Mientras no se ejecute la interrupción 100, es decir, en tanto no haya pasado 1 s, el TMR1 se vuelve a activar y se regresa al bucle *ADC*, retomándolo donde fue interrumpido.

Por su parte, al llegar a la rutina de interrupción 100, *RELOJ* se inicializa con el dato *0x63* para una nueva temporización de 1 s. Entonces, el tiempo establecido visualizado en el LCD, disminuye en uno. Esto significa, que en la pantalla de cristal líquido, se despliega el tiempo que resta del tiempo establecido en intervalos de 1 s. Enseguida, se reactiva el TMR1 y se retorna al bucle *ADC*, continuando con su ejecución en el lugar donde fue interrumpido.

Hay que recordar que, en el registro *ECLOCK*, se tiene la centésima de segundo del tiempo establecido; en *DCLOCK*, se localiza la décima de segundo del tiempo establecido; *BCLOCK* en tanto, almacena el valor de la unidad de minuto del tiempo establecido; y finalmente, en *ACLOCK*, se guarda la decena de minuto del tiempo establecido.

Cuando el tiempo establecido llega a cero, antes de salir de la interrupción, el bit 0 de *MASTER* se pone a ‘1’; se regresa entonces al bucle *ADC*, pero sin volver a habilitar el TMR1. En el instante en que dicho bucle inicie su secuencia, se comprueba si el bit 0 de *MASTER* es igual a ‘1’; como si es, se “rompe” el ciclo *ADC*, lo que nos permite salir de él, de la sección de código ‘PROCESO’ y, pasar a *ATLAST*, a la etapa de la electroválvula (figura 19).

Es importante mencionar que, la rutina de tratamiento de interrupción *INTER1*, permite que el tiempo establecido tenga un valor de hasta 90 minutos 00 segundos; es decir, se puede ingresar como máximo, un tiempo establecido de 90 minutos 00 segundos.

Con la interrupción *INTER1*, se logra que el tiempo establecido decremente su valor cada segundo, desplegando tal efecto en el LCD; esta rutina, se lleva a cabo dentro del bucle *ADC*, es decir, mientras se sensa la temperatura actual del horno de la termoformadora y se compara con la temperatura deseada. El microcontrolador PIC, trabaja con la tarea principal *ADC*, y responde a la interrupción *INTER1* cuando se produce; una vez atendida, se regresa a las tareas previas. Sin embargo, la acción es tan rápida al usuario, que pareciera que el PIC hiciera ambos procesos al mismo tiempo, simultáneamente. En esencia, esta es la base de la multitarea.

IV.4.6 Etapa de la electroválvula

Esta parte de PROYECTO.ASM, realiza la apertura y cierre de la electroválvula de la máquina termoformadora. También, nos indica un nuevo proceso de termoformado. Su código fuente, se encuentra en el Apéndice, páginas 75 a 77. En la figura siguiente, se tiene el diagrama de flujo respectivo.

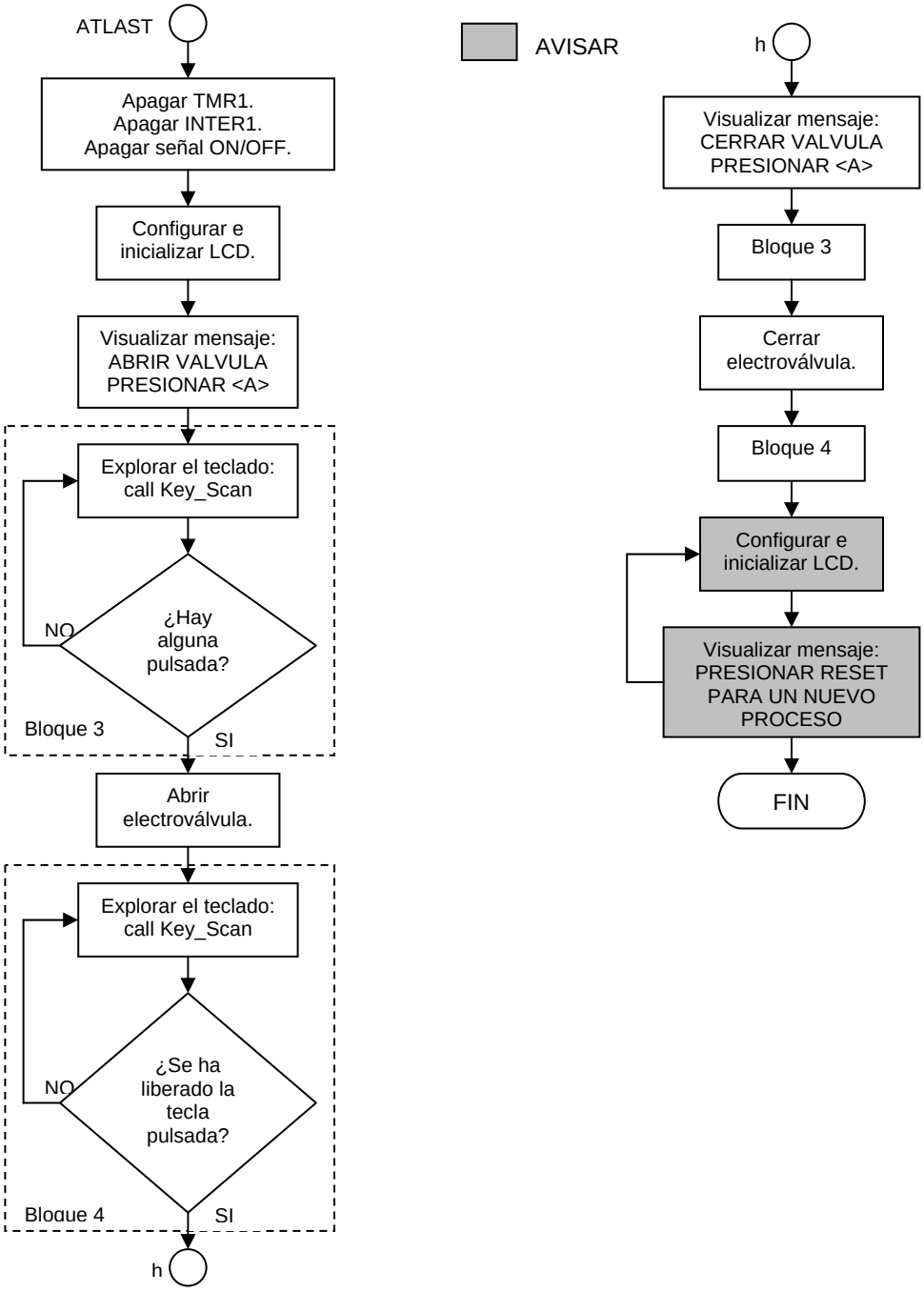

Figura 23. Diagrama de flujo de ATLAST.

El primer mensaje de este segmento, *ABRIR VALVULA PRESIONAR <A>*, nos indica que precisamente presionemos la tecla ‘A’ para abrir la electroválvula. Al llevar a cabo dicha acción, la bomba de vacío de la termoformadora succiona el aire del horno, provocando que la lámina de material termoplástico adquiera la forma del molde correspondiente.

Se despliega entonces, el segundo mensaje: *CERRAR VALVULA PRESIONAR <A>*; éste, nos avisa que presionemos la tecla ‘A’ una vez más para cerrar la electroválvula y en consecuencia, se deje de crear un vacío en el horno de la máquina de termoformado.

Una vez que se presiona la tecla en cuestión, se llega al bucle *AVISAR*. Este bucle, despliega en el LCD el mensaje: *PRESIONAR RESET PARA UN NUEVO PROCESO*. Mientras no se presione la tecla de RESET, este último mensaje se sigue visualizando. Si se desea realizar otro proceso de termoformado, entonces se debe de teclear RESET.

En este momento, el programa para el sistema de control de la temperatura de la termoformadora PROYECTO.ASM, llega a su fin. Si no se requiere de otro proceso de termoformado, se apaga el sistema de control.

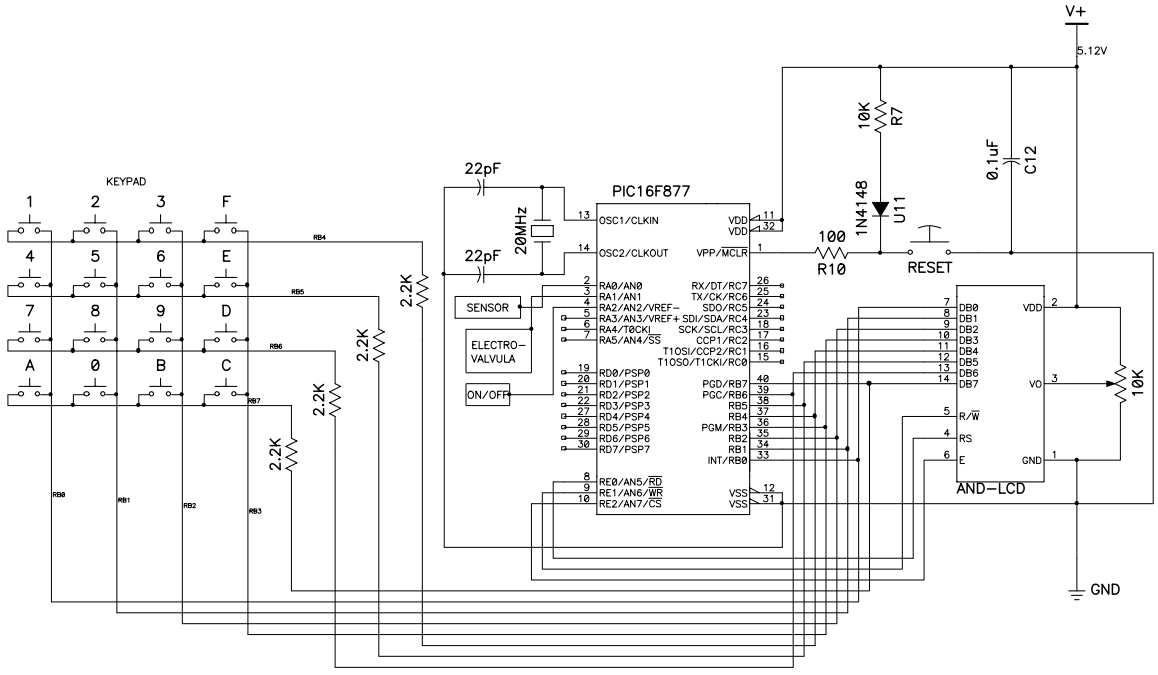

Figura 24. Diagrama esquemático de la etapa de control.

En la figura que se muestra en la página anterior, se tiene el diagrama esquemático de la etapa de control. A través del pin 0 del puerto A del microcontrolador PIC (RA0/AN0), se ingresa la señal analógica proveniente de la etapa del sensor-acondicionador. La salida, la señal de control ON/OFF, se entrega a la etapa de potencia por medio del pin 2 del puerto A (RA2/AN2/VREF-). Por su parte, el bit que abre y cierra la electroválvula, utiliza el pin 1 del puerto A (RA1/AN1) como salida.

V. ETAPA DE POTENCIA

V.1 INTRODUCCIÓN

El sistema de control de temperatura para una máquina de termoformado que se expone en el presente trabajo, es un sistema de control retroalimentado o de malla cerrada.

Este tipo de sistemas, cuenta con un sensor, el cual, sirve para medir el valor de la salida y con ello verificar si se consiguió el valor deseado. La señal que se obtiene del sensor, se compara con la señal de referencia y se determina la señal de error. El controlador recibe la señal de error, y genera una señal de control cuyo objetivo será reducir el error, o bien, mantenerlo en cero.

Se utiliza un controlador ON/OFF. Éste, junto con el comparador, se encuentran implementados en el microcontrolador PIC y forman el núcleo de la etapa de control, como se explicó en el capítulo anterior.

Entonces, de la etapa de control, específicamente, del pin 2 del puerto A del PIC16F877 (RA2/AN2/VREF-), se obtiene una señal ON/OFF. Una vez que se cuenta con dicha señal, se requiere de una etapa de potencia para adecuarla y así, poder controlar la temperatura del horno de la termoformadora en función de la señal de control.

La etapa de potencia se muestra en el siguiente diagrama a bloques:

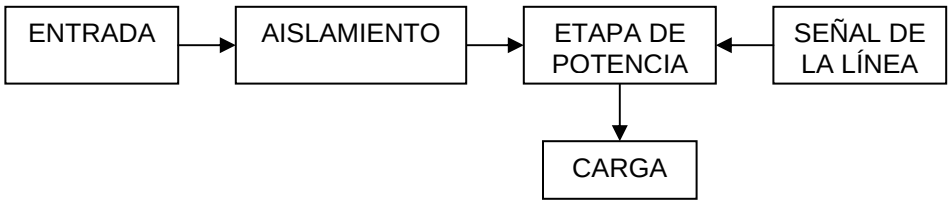

Figura 25. Diagrama a bloques de la etapa de potencia.

La entrada del sistema, es la señal proveniente del controlador ON/OFF. Esta señal de control o acción de control, es enviada a la etapa de potencia a través de un circuito de aislamiento, el cual protege a la parte digital (que trabaja en mili-Amperes) de la etapa de potencia (que opera en Amperes).

La etapa de potencia, está compuesta principalmente por un **TRIAC**, que es el dispositivo con el que se puede operar la señal del voltaje de línea (señal alterna). En esta etapa, es donde se conectan la señal de línea (127 VAC, 60 Hz) y la carga (horno de la termoformadora).

V.2 CIRCUITO DE AISLAMIENTO

La separación entre la etapa de control y la de potencia es muy importante, ya que el nivel de corriente que se trabaja en una y otra es diferente y, el estar unidos directamente puede provocar daños considerables e irreversibles al circuito de control; es necesario por lo tanto, un circuito de aislamiento. Se ocupa para esto, un **optoacoplador**.

Un optoacoplador, es un dispositivo semiconductor formado por un fotoemisor, un fotoreceptor y entre ambos, hay un camino por donde se transmite la luz. Todos estos elementos, se encuentran dentro de un encapsulado que por lo general es del tipo DIP.

La señal de entrada es aplicada al fotoemisor, y la salida, es tomada del fotoreceptor. Los optoacopladores, son capaces de convertir una señal eléctrica en una señal luminosa modulada y volver a convertirla en una señal eléctrica. No existe un contacto "físico" entre el emisor y el receptor, de ahí, el aislamiento eléctrico que se establece entre los circuitos de entrada y salida.

Los fotoemisores que se emplean en los optoacopladores, son diodos que emiten rayos infrarrojos (IRED) y los fotoreceptores, pueden ser tiristores o transistores.

Cuando aparece una tensión sobre las terminales del diodo IRED, éste emite un haz de rayos infrarrojo que se transmite a través de una pequeña guía de ondas de plástico o cristal hacia el fotoreceptor. La energía luminosa que incide sobre el fotoreceptor, hace que éste genere una tensión eléctrica a su salida.

De los diferentes tipos de optoacopladores que existen, se selecciona un **opto-TRIAC**, que no es más que un diodo emisor de luz (LED) acoplado ópticamente con un TRIAC fotodetector (foto-TRIAC), como se aprecia en la figura siguiente.

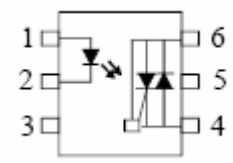

Figura 26. Esquema de un opto-TRIAC.

Al no ser un dispositivo que soporte grandes niveles de potencia, el propio foto-TRIAC muchas veces actúa sobre el control de un TRIAC de mucho mayor potencia, como es nuestro caso.

El opto-TRIAC que se utiliza, es el **MOC3032**. Se trata de un opto-TRIAC con detección de cruce por cero, un optoacoplador en cuya etapa de salida se encuentra un TRIAC de cruce por cero. Esta característica, da la ventaja de poder conmutar cargas grandes con dispositivos no tan potentes.

Pensemos en el interruptor de una habitación, por ejemplo. Si se acciona puede ser que salga alguna chispa, pero, si se pudiera activar justo en el momento en el que el voltaje es cero, la intensidad del circuito en aquel momento también sería cero. Cuando la corriente comenzara a aumentar, el interruptor ya estaría accionado y no habría chispa.

Aplicando lo anterior a nuestra etapa de potencia, significa que una vez que el opto-TRIAC recibe la señal de control, esperará a que la señal alterna de la línea cruce por cero para entonces, dar su disparo y accionar así al TRIAC de potencia. Con ello, se previene que el circuito presente interferencia, perturbaciones y/o ruido, al interrumpirse la señal en cualquier otro punto diferente de cero.

V.3 EL TRIAC

Un TRIAC (TRIode for Alternative Current), es un SCR bidireccional que se comporta como dos SCR´s en paralelo e invertidos, de tal manera que este dispositivo puede controlar corriente en cualquier dirección. Normalmente, tiene una tensión de ruptura alta y el procedimiento de hacerlo entrar en conducción es a través de un pulso de disparo de puerta (positivo o negativo). En la figura 27, se muestra su símbolo (a) y el modelo equivalente basado en dos SCR´s conectados en oposición (b).

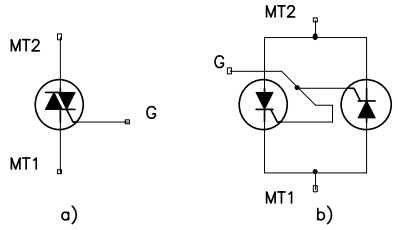

Figura 27. a) Símbolo del TRIAC y b) Modelo equivalente en SCR´s.

El TRIAC empleado en el circuito de potencia es el **NTE5695**. Las características principales a considerar en el control de este dispositivo son:

- $I_{T(RMS)}$, la corriente rms en el estado de conducción, la corriente máxima que puede pasar a través del TRIAC. Para el NTE5695, $I_{T(RMS)}$= 40 A.
- V_{DRM}, voltaje de pico repetitivo en estado de bloqueo, voltaje de transición conductiva. En el caso del NTE5695, V_{DRM}= 400 V. Si el voltaje instantáneo

aplicado de MT2 a MT1 llegara a rebasar este valor, el TRIAC transita al estado conductivo provocando una pérdida de control.

- I_{GT}, la corriente de la compuerta (Gate). En el NTE5695, I_{GT} tiene un valor máximo de 50 mA.

En la figura 28, se presenta el diagrama esquemático de la etapa de potencia. Se usa esta configuración, debido a que la carga que el circuito soporta es del tipo resistivo. En caso de tener componentes en la carga que almacenen energía, se emplea otro tipo de conexión.

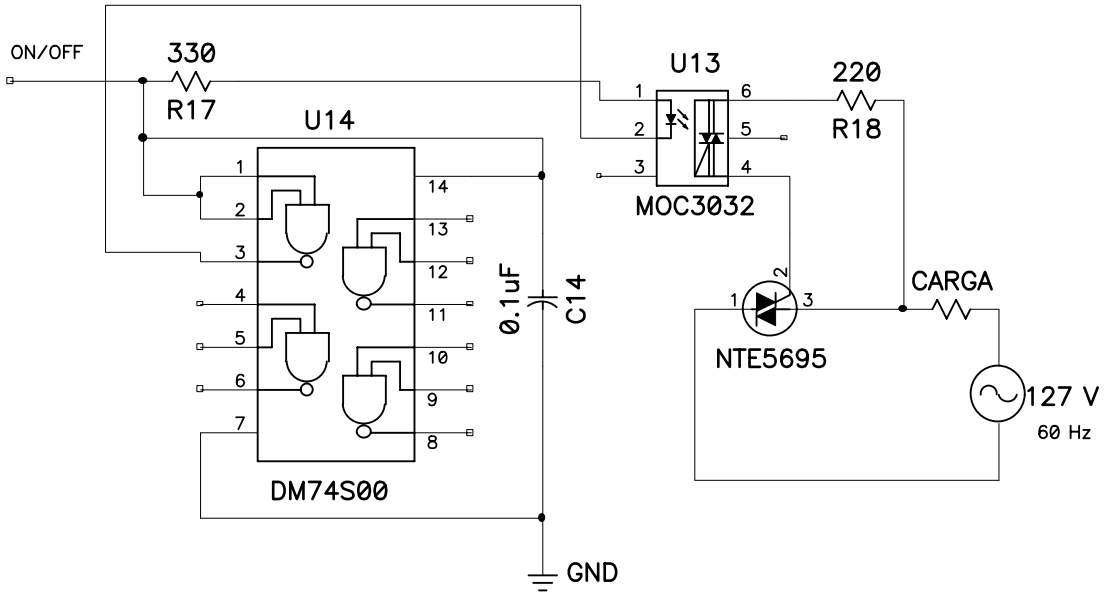

Figura 28. Diagrama esquemático de la etapa de potencia.

La señal de control llega al DM74S00, que es un circuito integrado con cuatro compuertas NAND de dos entradas cada una. Una de estas compuertas, se configura como inversor; la salida de éste, proporciona un '0' lógico por lo que circulará corriente a través del diodo emisor perteneciente al MOC3032, activándolo. El LED prende entonces al foto-TRIAC, que a su vez enciende al TRIAC de potencia NTE5695 y de este modo, la señal de alterna empezará a alimentar al horno de la máquina de termoformado.

La resistencia de entrada R17, limita la corriente que pasa a través del diodo emisor de luz del optoacoplador; esta corriente debe de tener como mínimo 10 mA, pero no más de 50 mA. 15 mA es un valor recomendable, el cual, asegura una larga vida de operación del optoacoplador. Una corriente mayor a 15 mA, no mejora el desempeño del MOC, por el contrario, puede provocar que el deterioro inherente en los LED´s sea más rápido.

Asumiendo una caída de voltaje de 1.5 V en el LED del optoacoplador en 15 mA, permite una simple fórmula para calcular el resistor de entrada R17:

$$R17 = \frac{Vcc - 1.5}{0.015}$$

Sabiendo que el nivel de voltaje de los pulsos de la señal de control ON/OFF es de 5 V, entonces:

$$R17 = \frac{5 - 1.5}{0.015} = 233.33\Omega$$

Si seleccionamos una resistencia con un valor de 220Ω, la corriente sería de 15.91 mA. Se elige un resistor de 330Ω, con el cual, la corriente que circula por el LED es de 10.61 mA, un valor dentro del rango aceptado.

En cuanto a la resistencia R18, no es obligatorio su uso cuando la carga es resistiva (como es el caso), debido a que la corriente es limitada por I_{GT} del TRIAC de potencia. Sin embargo, este resistor previene una posible destrucción del optoacoplador en aplicaciones donde la carga es altamente inductiva.

R18 se calcula con la ecuación: $R18(\min) = Vin(pk)/1.2A$

Como estamos operando un voltaje nominal de 127 VAC, se considera un pico de voltaje de línea de V_{in}(pk)= 180 Vac. Luego:

$$R18(\min) = 180/1.2 = 150\Omega$$

Se escoge una R18 con un valor de 220Ω, para asegurar que el circuito no tendrá problemas por corrientes excesivas.

El TRIAC, pertenece a los dispositivos semiconductores llamados tiristores. Éstos, son sumamente populares en el control de potencia en cargas resistivas e inductivas como motores, solenoides, calefactores, etc. Comparados con sus contrapartes mecánicos como son los relés, los tiristores ofrecen una mayor fiabilidad, duración, velocidad y un menor costo.

Las técnicas de aislamiento, están basadas en transformadores u optoacopladores. Se elige la segunda por dos motivos: direccionalidad y prestaciones. Un optoacoplador es unidireccional, la señal va en un único sentido a diferencia de un transformador, que es bidireccional. Además, presenta mayores prestaciones desde el punto de vista de costo, volumen y fiabilidad. El optoacoplador que se usa, al tener la característica de detección de cruce por cero, sincroniza la señal de control digital y la fase de la línea. La falta de ésta, de sincronización, afecta fuertemente a la carga y a los dispositivos de potencia, que reducen su vida media de duración.

VI. SISTEMA DE CONTROL DE TEMPERATURA

Recordando, un sistema de control es aquel en el cual podemos manipular o ajustar a voluntad alguna característica o propiedad. Dicha característica o propiedad no es otra cosa que alguna variable física, por ejemplo: posición, velocidad, aceleración, presión, gasto, temperatura, nivel; por mencionar las más utilizadas. En nuestro caso, se trata de la temperatura del horno de la máquina de termoformado que se encuentra en el Laboratorio de Ingeniería de Producto, en el CCADET-UNAM.

El sistema de control de temperatura desarrollado en esta tesis, es del tipo retroalimentado o de malla cerrada y se basa en un microcontrolador PIC. En la figura 36, en la página 84 del Apéndice, se muestra físicamente dicho sistema.

El control de temperatura consiste de un termopar tipo K como sensor de temperatura, una etapa de acondicionamiento de la señal proveniente del sensor, un controlador ON/OFF que junto con el comparador se encuentran implementados vía software en un microcontrolador PIC y forman el núcleo de la etapa de control, esta etapa incluye un display de cristal líquido (LCD) de 2x16 y un teclado matricial de 4x4 hexadecimal, y finalmente de una etapa de potencia. En la figura 37, en la página 85 del Apéndice, se observa el circuito impreso de control así como el interior del gabinete que lo contiene.

Ahora, el procedimiento para utilizar el sistema de control de temperatura desarrollado es el que se indica a continuación:

Al encender el equipo, en el LCD aparece el mensaje *TEMP(°C):* , indicándonos que debemos de ingresar al sistema la temperatura de operación deseada del horno de la máquina de termoformado, a través del teclado; dicha temperatura de referencia puede tener cualquier valor entre 0 y 240 °C. En la figura 29 se tiene el despliegue correspondiente.

Figura 29. Mensaje de ingreso de la temperatura de referencia.

Si por alguna razón, nos equivocamos al momento de teclear la temperatura requerida, basta con presionar el botón de RESET, inicializando el sistema y apareciendo nuevamente el mensaje *TEMP(°C):* .

En cuanto al ingreso del valor de la temperatura, éste debe de constar de 3 dígitos para todo el rango. Ver figura 30.

Figura 30. Ejemplo de formato para el valor de la temperatura de referencia.

La pantalla de cristal líquido está lista en este momento, para desplegar el segundo mensaje del programa, *TIME(min):* ; éste, nos indica que ingresemos (a través del teclado) el tiempo establecido, es decir, el tiempo que el horno de la máquina de termoformado permanecerá encendido a la temperatura requerida, en el intervalo de 0 a 10 minutos. En la figura 31, se observa el despliegue correspondiente.

Figura 31. Mensaje de ingreso del tiempo requerido.

Si por alguna razón, nos equivocamos al momento de teclear el tiempo requerido, basta con presionar el botón de RESET, inicializando el sistema y

apareciendo nuevamente el mensaje *TEMP(°C):* . Se teclea otra vez el valor de la temperatura de referencia y enseguida aparece el mensaje *TIME(min):* .

El valor del tiempo establecido para todo el intervalo, tiene el formato ##.##; es decir, para un tiempo establecido de 9 minutos 25 segundos por ejemplo, primero se teclea 09 y luego 25. Los minutos pueden ser 00, 01, 02… hasta 10; los segundos en tanto, pueden ser 00, 01, 02… hasta 59.

Con los datos de entrada, temperatura de referencia y tiempo establecido, especificados, el programa tiene lo necesario para iniciar el proceso de control de temperatura. En el LCD se observa, que en la parte izquierda de la segunda línea aparece un valor de temperatura; éste, es la temperatura actual del horno de la máquina de termoformado, la temperatura medida proveniente del termopar. Mientras tanto, en la parte derecha de la segunda línea, queda sólo el valor del tiempo establecido. Ver figura 32.

Figura 32. Ejemplo de temperatura sensada del horno de la máquina de termoformado y tiempo establecido.

El microcontrolador utiliza la temperatura deseada (referencia) y la temperatura medida, para generar una señal de control de tipo ON-OFF. Ambas temperaturas, se comparan continuamente; si la primera es superior a la segunda, se envía una señal de control (un bit) a través de una de las líneas de E/S del microcontrolador hacia la etapa de potencia, la cual a su vez activa la alimentación del horno para aumentar su temperatura interior.

Por el contrario, cuando la segunda es igual o mayor a la primera, se desactiva la señal ON/OFF y por ende, se apaga la alimentación del horno de la termoformadora.

La primera vez que la temperatura actual del horno es igual o mayor a la de referencia, el tiempo establecido empieza a decrementar su valor, mientras continúa la comparación de las temperaturas.

La temperatura actual del horno así como el tiempo transcurrido, aparecen en todo momento en el display, al igual que la temperatura de referencia.

El ciclo de comparación se realiza una y otra vez, apagando o encendiendo el horno de la máquina de termoformado en base a la señal de control. Sin embargo, en algún momento debe de terminar y esto es, cuando el tiempo establecido llega a cero.

El LCD entonces, nos muestra un nuevo mensaje: *ABRIR VALVULA PRESIONAR <A>*.

Este mensaje nos señala, que debemos presionar la tecla 'A' para abrir la electroválvula. Al llevar a cabo tal acción, la bomba de vacío de la termoformadora succiona el aire del horno, provocando que la lámina de material termoplástico adquiera la forma del molde correspondiente. En la figura siguiente, se tiene el despliegue en cuestión.

Figura 33. Mensaje para abrir la electroválvula.

Una vez realizado la acción anterior, se despliega a continuación, el mensaje: *CERRAR VALVULA PRESIONAR <A>*; éste, nos avisa que presionemos la tecla 'A' una vez más para cerrar la electroválvula y en consecuencia, se deje de crear un vacío en el horno de la máquina de termoformado. Ver figura 34.

Figura 34. Mensaje para cerrar la electroválvula.

Con la electroválvula cerrada, aparece finalmente en el LCD el mensaje *PRESIONAR RESET PARA UN NUEVO PROCESO*, indicándonos lo que hay que hacer si se requiere realizar otro proceso de termoformado. En la figura que sigue, se muestra el mensaje correspondiente.

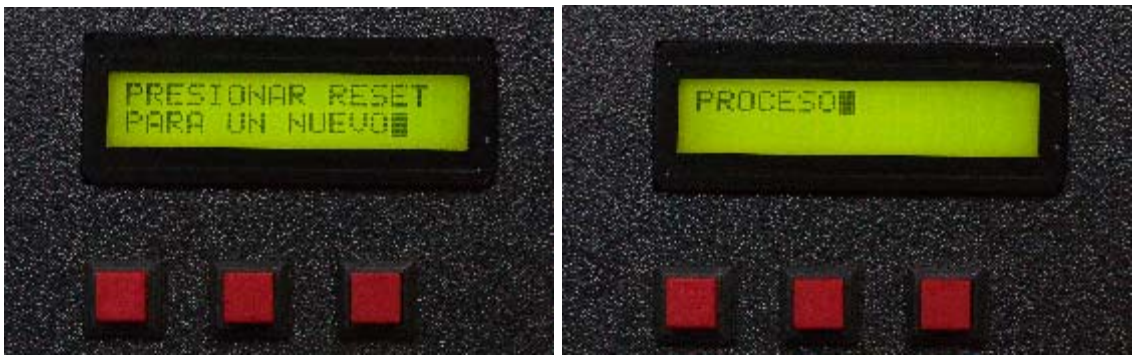

Figura 35. Mensaje para llevar a cabo otro proceso de termoformado.

En este momento, el programa para el sistema de control de la temperatura de la termoformadora, llega a su término. Si no se requiere de otro proceso de termoformado, se apaga el sistema de control.

Cabe hacer un par de observaciones con respecto al sistema de control de temperatura:

Como se explicó en el capítulo 4 sección IV.4.1, el teclado que se utiliza es del tipo matricial 4x4 hexadecimal. Sin embargo, no se utilizan las 16 teclas; sólo las numéricas (0-9) y la letra ‘A’.

Entonces, para evitar que el usuario presione por accidente una tecla no utilizada (B, C, D, E y F) al momento de ingresar la temperatura de referencia y/o el tiempo establecido, dichas teclas físicamente no están conectadas a ningún botón, aunque en el programa sí se contemplan. Lo anterior se dejó así, por si en un futuro se requieren tales letras.

Por otro lado, si se desea cancelar el proceso de termoformado, solo basta con presionar el botón de RESET; no importa en que parte del proceso nos encontremos.

VII. CONCLUSIONES

Con el desarrollo del presente trabajo, las características con las cuales se pidió el sistema de control de temperatura, un rango de temperatura de 0 a 240 °C y un intervalo de tiempo de 0 a 10 minutos, se cumplieron completamente. Aún más, dichas características se mejoraron, quedando en realidad el sistema con un rango de temperatura de 0 a 255 °C y un intervalo de tiempo de 0 a 90 minutos.

El control de temperatura es accesible y fácil de manejar, para lo cual, se incorporó un display LCD donde se puede observar en todo momento, el comportamiento del sistema. Así también, se dispuso de un teclado para el ingreso de datos.

Dicho teclado, físicamente, solo consta de los números del 0 al 9, a pesar de ser hexadecimal; sin embargo, en el programa de control sí se consideran las letras de la 'A' a la 'F' aunque no se utilicen (excepto la letra 'A'). Lo anterior se realizó de esta manera, pensando en que en un futuro sí se usen.

Este trabajo puede tener muchas aplicaciones, ya que la temperatura es una variable presente en muchos procesos industriales y es necesario controlarla. La fluctuación máxima del sistema de control desarrollado es de ± 2 °C, en el intervalo especificado por lo que, por ejemplo, se puede utilizar para controlar un horno industrial de secado; el cual a su vez, se podría usar entonces como cámara de reacción a temperatura controlada para la síntesis de zeolitas y películas delgadas o, como horno a temperatura controlada para los reactores de fluoración de diversos materiales.

Lo realizado en la presente tesis, puede servir como punto de partida para diseñar sistemas de control de temperatura más complejos. Como se mencionó en el párrafo anterior, la desviación máxima de la temperatura medida con respecto a la de referencia es de ± 2 °C. Hay que recordar que el control de temperatura realizado es del tipo ON-OFF, así pues, dicha desviación se puede reducir si se implementa otro tipo de controlador, por ejemplo un PID.

Más aún, podemos aprovechar la capacidad del microcontrolador PIC para establecer comunicación con una computadora y así, en lugar de utilizar un teclado matricial y un display LCD, se podría ocupar el teclado y el monitor de la PC.

VIII. BIBLIOGRAFÍA

- Amplificadores Operacionales y Circuitos Integrados Lineales.
  Coughlin, Robert y Driscoll, Frederick.
  Editorial Prentice Hall Hispanoamericana S.A., 1ª edición, México, 1993.

- Applications of Non Zero Crossing Triac Drivers Featuring the MOC3011.
  Fairchild Semiconductor.
  Application Note AN-3003, 2003.

- Applications of Zero Voltage Crossing Optically Isolated Triac Drivers.
  Fairchild Semiconductor.
  Application Note AN-3004, 2002.

- BURR-BROWN Integrated Circuits Data Book. Volume 33.
  Burr-Brown Corporation.
  USA, 1989.

- Circuitos con Triacs, Diacs y Tiristores.
  Dr. Ing. Bergtold, Fritz.
  Editorial Gustavo Gili S.A., 1985.

- Design with Operational Amplifiers and Analog Integrated Circuits.
  Franco, Sergio.
  McGraw-Hill Book Company, International edition, Singapore, 1988.

- Linear Circuits Data Book. Volume 1.
  Texas Instruments Inc.
  USA, 1989.

- Manual Técnico TERMOFORMADO.
  Plastiglas de México, S.A. de C.V.
  2003.

- MICROCONTROLADORES PIC. Diseño práctico de aplicaciones.
  Primera parte: PIC 16F84.
  Angulo, J. Ma. y Angulo, I.
  Editorial McGraw-Hill/Interamericana, 2ª edición, España, 2000.

- MICROCONTROLADORES PIC. Diseño práctico de aplicaciones.
  Segunda parte: PIC 16F87X.
  Angulo, J. Ma., Romero, S. y Angulo, I.
  Editorial McGraw-Hill/Interamericana, 2ª edición, España, 2000.

- MICROCONTROLADORES PIC. La solución en un chip.
  Angulo, J. Ma., Martín, E. y Angulo, I.
  Editorial ITP Paraninfo, 4ª edición, España, 2000.

- MPASM Assembler User's Guide.
  Microchip Technology Incorporated.
  USA, 1996.

- National Semiconductor's Temperature Sensor Handbook.
  National Semiconductor.
  1999.

- NTE's Technical Guide and Cross Reference.
  NTE Electronics, Inc.
  5th edition, USA, 1989.

- PIC16F87X Data Sheet.
  Microchip Technology Inc.
  2001.

- Temperature Sensing Technologies.
  Baker, Bonnie.
  Microchip Technology Inc., AN679, 2002.

- THERMOCOUPLE MEASUREMENT.
  Williams, Jim.
  Linear Technology, Application Note 28, 1988.

- Single Supply Temperature Sensing with Thermocouples.
  Baker, Bonnie.
  Microchip Technology Inc., AN684, 2002.

- Sistemas Realimentados de Control. Análisis y Síntesis.
  D'Azzo.
  Editorial Paraninfo.

IX. APÉNDICE

IX.1 CÓDIGO FUENTE DEL PROGRAMA DEL SISTEMA PROPUESTO

```
;Programa PROYECTO.ASM

            LIST        P=16F877        ;Se indica el tipo de microcontrolador
            RADIX       HEX             ;Sistema de numeración hexadecimal
            INCLUDE     <P16F877.INC>   ;Se incluye la definición de los registros
                                        ;internos en una librería

;Desliga el mensaje de aviso N° 302:
;"Register in operand not in bank 0. Ensure that bank bits are correct"

            ERRORLEVEL      -302

Lcd_var     EQU     0x20    ;Inicio de las variables para el LCD
Key_var     EQU     0x22    ;Inicio de las variables del teclado
TEMPO_1     EQU     0x29    ;Variable temporal 1
TEMPO_2     EQU     0x2A    ;Variable temporal 2
CONTA_C     EQU     0x2B    ;Contador de caracteres
CONTA_1     EQU     0x2C    ;Contador de caracteres 1
CUENTA      EQU     0x2D    ;Contador de caracteres del tiempo
BASE        EQU     0x2E    ;Constante que permite almacenar
                            ;la temperatura de referencia
                            ;y el tiempo establecido
TRBCD0      EQU     0x2F    ;Variable más significativa de
                            ;la temperatura de referencia
TRBCD1      EQU     0x30    ;Variable intermedia de la
                            ;temperatura de referencia
TRBCD2      EQU     0x31    ;Variable menos significativa de
                            ;la temperatura de referencia
ACQUIRE     EQU     0x32    ;Temporizador para adquisición del
                            ;módulo A/D
DIFER_H     EQU     0x33    ;Parte alta del valor binario obtenido
                            ;del módulo ADC
DIFER_L     EQU     0x34    ;Parte baja del valor binario obtenido
                            ;del módulo ADC
COCIENT     EQU     0x35    ;Almacena el resultado de la división
CODIGOH     EQU     0x36    ;Parte alta del resultado de la conversión
                            ;de binario a BCD
CODIGOL     EQU     0x37    ;Parte baja del resultado de la conversión
                            ;de binario a BCD
BODEGA      EQU     0x38    ;Registro temporal
LAPSO       EQU     0x39    ;Registro temporal para obtener la unidad
RELOJ       EQU     0x3A    ;Temporizador de 1 s para la interrupción
ACLOCK      EQU     0x3B    ;Registro que almacena la decena de minuto
                            ;del tiempo establecido
BCLOCK      EQU     0x3C    ;Registro que almacena la unidad de minuto
                            ;del tiempo establecido
DCLOCK      EQU     0x3D    ;Registro que almacena la décima de segundo
                            ;del tiempo establecido
ECLOCK      EQU     0x3E    ;Registro que almacena la centésima de segundo
                            ;del tiempo establecido
MASTER      EQU     0x3F    ;Registro que permite salir del ciclo de
```

```
                                        ;conversión y pasar a la etapa de la
                                        ;electroválvula
CERO        EQU         0x40            ;Registro igual a cero
W_TEMP      EQU         0x41            ;Registro temporal de W
STATUST     EQU         0x42            ;Registro temporal de STATUS
PCLATHT     EQU         0x43            ;Registro temporal de PCLATH

            ORG         0x00            ;Inicio en el Vector de Reset
            goto        INICIO          ;Va a la primera instrucción
                                        ;del programa
            ORG         0x04            ;Vector de Interrupción
            goto        INTER1

            INCLUDE     <TECLADO.INC>   ;Incluye rutinas de manejo del teclado

TABLA_M     movwf       PCL             ;Desplazamiento sobre la tabla

MENS_0      EQU         $               ;MENS_0 apunta al primer caracter
            retlw       0x54            ;T
            retlw       0x45            ;E
            retlw       0x4D            ;M
            retlw       0x50            ;P
            retlw       0x28            ;(
            retlw       0xDF            ;°
            retlw       0x43            ;C
            retlw       0x29            ;)
            retlw       0x3A            ;:
            retlw       ' '             ;Espacio en blanco
            retlw       0x00            ;Último caracter del mensaje 0

MENS_1      EQU         $               ;MENS_1 apunta al primer caracter
            DT          "TIME(min):",0x00

MENS_2      EQU         $               ;MENS_2 apunta al primer caracter
            DT          "ABRIR VALVULA",0x00

MENS_3      EQU         $               ;MENS_3 apunta al primer caracter
            DT          "PRESIONAR <A>",0x00

MENS_4      EQU         $               ;MENS_4 apunta al primer caracter
            DT          "CERRAR VALVULA",0x00

MENS_5      EQU         $               ;MENS_5 apunta al primer caracter
            DT          "PRESIONAR RESET",0x00

MENS_6      EQU         $               ;MENS_6 apunta al primer caracter
            DT          "PARA UN NUEVO",0x00

MENS_7      EQU         $               ;MENS_7 apunta al primer caracter
            DT          "PROCESO",0x00

            INCLUDE     <LCD_CXX.INC>   ;Incluye rutinas de manejo del LCD
```

```
;*******************************************************************************
;MENSAJE: Esta rutina visualiza en el LCD el mensaje cuyo inicio está indicado en
;el acumulador. El fin de un mensaje se determina mediante el código 0x00

MENSAJE     movwf     TEMPO_1          ;Salva posición de la tabla
MENSAJ1     movf      TEMPO_1,0        ;Recupera posición de la tabla
            call      TABLA_M          ;Busca caracter de salida
            movwf     TEMPO_2          ;Guarda el caracter
            movf      TEMPO_2,1
            btfss     STATUS,2         ;Mira si es el último
            goto      NO_ES            ;No es el último
            return                     ;Sí es el último
NO_ES       call      LCD_DATO         ;Visualiza en el LCD
            incf      TEMPO_1,1        ;Siguiente caracter
            goto      MENSAJ1
;*******************************************************************************
;TEMPR: Esta rutina almacena cada tecla pulsada de la temperatura de referencia
;en los registros TRBCD0, TRBCD1 y TRBCD2, en código BCD

TEMPR       movf      Tecla,0          ;Carga Tecla en W
            btfsc     BASE,0           ;¿Es la primera pulsada?
            movwf     TRBCD0           ;Sí. Almacena dígito más significativo
            btfss     BASE,0           ;No. ¿Es la tercera pulsada?
            movwf     TRBCD2           ;Sí. Almacena dígito menos significativo
            btfss     BASE,1           ;No. ¿Es la segunda pulsada?
            movwf     TRBCD1           ;Sí. Almacena el dígito intermedio
            rrf       BASE,1           ;No
            return
;*******************************************************************************
;TIMEREF: Esta rutina almacena cada tecla pulsada del tiempo establecido en los
;registros ACLOCK, BCLOCK, DCLOCK y ECLOCK, en código BCD

TIMEREF     movf      Tecla,0          ;Carga Tecla en W
            movwf     ECLOCK           ;Almacena la centésima de segundo
            btfsc     BASE,5           ;¿Es la última pulsada?
            goto      SIES             ;Sí
            movf      ECLOCK,0         ;No. Carga ECLOCK en W
            btfsc     BASE,0           ;¿Es la primera pulsada?
            movwf     ACLOCK           ;Sí. Almacena la decena de minuto
            btfss     BASE,0           ;No. ¿Es la tercera pulsada?
            movwf     DCLOCK           ;Sí. Almacena la décima de segundo
            btfss     BASE,1           ;No. ¿Es la segunda pulsada?
            movwf     BCLOCK           ;Sí. Almacena la unidad de minuto
            rrf       BASE,1           ;No
SIES        return
;*******************************************************************************
;DIV: Esta rutina divide el resultado de la conversión A/D, almacenado en los registros
;DIFER_H y DIFER_L, por 2 por medio de un corrimiento a la derecha un lugar

DIV         clrf      COCIENT          ;Limpia el resultado de la división
            rrf       DIFER_H,1        ;Divide el resultado de la conversión A/D por
            rrf       DIFER_L,0        ;2 y lo guarda temporalmente en W
            movwf     COCIENT          ;Resultado de la división
            return
;*******************************************************************************
```

```
;B_BCD: Esta rutina convierte el valor binario localizado en COCIENT en BCD. El
;resultado se deposita en las variables CODIGOH y CODIGOL

B_BCD       movf        COCIENT,0       ;Carga el valor binario inicial
            clrf        CODIGOL         ;Inicia registros de trabajo
            clrf        CODIGOH
B_BCD1      addlw       0xF6            ;Resta 10 mediante suma de complemento a 2
            btfss       STATUS,0        ;¿Hay acarreo?
            goto        B_BCD3          ;No
            movwf       BODEGA          ;Sí. Guardar en registro temporal
            incf        CODIGOL,1       ;Incrementar byte bajo, guardarlo y enviar
            movf        CODIGOL,0       ;copia a W
            xorlw       b'00001010'     ;0Ah xor W
            btfss       STATUS,2        ;¿CODIGOL es igual a 10?
            goto        B_BCD2          ;No
            clrf        CODIGOL         ;Sí. Limpiar CODIGOL
            incf        CODIGOH,1       ;Incrementar CODIGOH
B_BCD2      movf        BODEGA,0        ;Recupera el dato
            goto        B_BCD1          ;Continuar la operación
B_BCD3      addlw       0x0A
            swapf       CODIGOL,1       ;<3:0> <==> <7:4> y ==> CODIGOL
            iorwf       CODIGOL,1       ;W or CODIGOL ==> CODIGOL
            return
;*****************************************************************************************

;Rutina de tratamiento de la interrupción

INTER1      movwf       W_TEMP          ;Copia W en su registro temporal
            swapf       STATUS,0        ;Swap STATUS y lo carga en W
            clrf        STATUS          ;Banco 0
            movwf       STATUST         ;Copia STATUS en su registro temporal
            movf        PCLATH,0        ;Carga PCLATH en W
            movwf       PCLATHT         ;Copia PCLATH en su registro temporal
            clrf        PCLATH          ;Página 0
            bcf         PIR1,0          ;Repone flag del TMR1
            clrf        T1CON           ;Desactiva el TMR1
            movlw       0xB0            ;Valor a partir del cual el TMR1
            movwf       TMR1L           ;empieza la temporización para
            movlw       0x3C            ;lograr una interrupción cada
            movwf       TMR1H           ;10 ms
            decfsz      RELOJ,1         ;¿Ha transcurrido 1 s?
            goto        SALIDA          ;No
            movlw       0x63            ;Sí. Carga el RELOJ para una temporización
            movwf       RELOJ           ;de 1 s
            movf        ECLOCK,0        ;¿La centésima de segundo
            xorwf       CERO,0          ;del tiempo establecido
            btfsc       STATUS,2        ;es igual a cero?
            goto        LOOK0           ;Sí
            decf        ECLOCK,1        ;No
            call        CHECARE         ;Reconfigura el LCD
            movf        ECLOCK,0        ;Ajuste ASCII de los caracteres
            addlw       0x30            ;del 0 al 9
            call        LCD_DATO        ;Visualiza la centésima de segundo del
                                        ;tiempo establecido
            goto        SALIDA
LOOK0       movf        DCLOCK,0        ;¿La décima de segundo del
```

```
        xorwf   CERO,0          ;tiempo establecido es
        btfsc   STATUS,2        ;igual a cero?
        goto    LOOK1           ;Sí
        decf    DCLOCK,1        ;No
        call    CHECARD         ;Reconfigura el LCD
        movf    DCLOCK,0        ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza la décima de segundo del
                                ;tiempo establecido
        movlw   0x09            ;Carga a ECLOCK con el número 9
        movwf   ECLOCK
        call    CHECARE         ;Reconfigura el LCD
        movf    ECLOCK,0        ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza la centésima de segundo del
                                ;tiempo establecido
        goto    SALIDA
LOOK1   movf    BCLOCK,0        ;¿La unidad de minuto del
        xorwf   CERO,0          ;tiempo establecido es
        btfsc   STATUS,2        ;igual a cero?
        goto    LOOK2           ;Sí
        decf    BCLOCK,1        ;No
        call    CHECARB         ;Reconfigura el LCD
        movf    BCLOCK,0        ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza la unidad de minuto del
                                ;tiempo establecido
        movlw   0x05            ;Carga a DCLOCK con el número 5
        movwf   DCLOCK
        call    CHECARD         ;Reconfigura el LCD
        movf    DCLOCK,0        ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza la décima de segundo del
                                ;tiempo establecido
        movlw   0x09            ;Carga a ECLOCK con el número 9
        movwf   ECLOCK
        call    CHECARE         ;Reconfigura el LCD
        movf    ECLOCK,0        ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza la centésima de segundo del
                                ;tiempo establecido
        goto    SALIDA
LOOK2   movf    ACLOCK,0        ;¿La decena de minuto del
        xorwf   CERO,0          ;tiempo establecido es
        btfsc   STATUS,2        ;igual a cero?
        goto    LOOK3           ;Sí
        decf    ACLOCK,1        ;No
        call    CHECARA         ;Reconfigura el LCD
        movf    ACLOCK,0        ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza la decena de minuto del
                                ;tiempo establecido
        movlw   0x09            ;Carga a BCLOCK con el número 9
        movwf   BCLOCK
        call    CHECARB         ;Reconfigura el LCD
        movf    BCLOCK,0        ;Ajuste ASCII de los caracteres
```

```
                addlw   0x30            ;del 0 al 9
                call    LCD_DATO        ;Visualiza la unidad de minuto del
                                        ;tiempo establecido
                movlw   0x05            ;Carga a DCLOCK con el número 5
                movwf   DCLOCK
                call    CHECARD         ;Reconfigura el LCD
                movf    DCLOCK,0        ;Ajuste ASCII de los caracteres
                addlw   0x30            ;del 0 al 9
                call    LCD_DATO        ;Visualiza la décima de segundo del
                                        ;tiempo establecido
                movlw   0x09            ;Carga a ECLOCK con el número 9
                movwf   ECLOCK
                call    CHECARE         ;Reconfigura el LCD
                movf    ECLOCK,0        ;Ajuste ASCII de los caracteres
                addlw   0x30            ;del 0 al 9
                call    LCD_DATO        ;Visualiza la centésima de segundo del
                                        ;tiempo establecido
                goto    SALIDA
LOOK3           bsf     MASTER,0        ;Pone a 1 el bit 0 de MASTER
                movf    PCLATHT,0       ;Restaura PCLATH
                movwf   PCLATH
                swapf   STATUST,0       ;Restaura STATUS
                movwf   STATUS
                swapf   W_TEMP,1        ;Restaura W
                swapf   W_TEMP,0
                retfie                  ;Salir de la interrupción
SALIDA          movf    PCLATHT,0       ;Restaura PCLATH
                movwf   PCLATH
                swapf   STATUST,0       ;Restaura STATUS
                movwf   STATUS
                swapf   W_TEMP,1        ;Restaura W
                swapf   W_TEMP,0
                bsf     T1CON,0         ;Activa el TMR1
                retfie                  ;Salir de la interrupción

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

;Comienzo del programa principal

INICIO          clrf    PORTB           ;Limpia salidas
                clrf    PORTE
                bsf     STATUS,5        ;Cambio al banco 1
                bcf     STATUS,6
                movlw   0x8E            ;Configura todos los pines del
                movwf   ADCON1          ;puerto E como E/S digitales
                clrf    TRISB           ;Puertos B y E como
                clrf    TRISE           ;salida
                bcf     STATUS,5        ;Cambio al banco 0
                call    UP_LCD          ;Configuración de líneas para el LCD
                call    LCD_INI         ;Inicialización del LCD

;Salida del mensaje "TEMP(°C): "

                movlw   MENS_0          ;Visualiza el mensaje
                call    MENSAJE
                movlw   .3              ;Inicia contador de
```

```
                movwf   CONTA_C         ;caracteres
                movlw   b'00011001'     ;Constante que permite almacenar
                movwf   BASE            ;la temperatura de referencia

LOOP            call    Key_Scan        ;Explora el teclado
                movlw   0x80
                subwf   Tecla,0
                btfsc   STATUS,2        ;¿Hay alguna pulsada?
                goto    LOOP            ;No. Seguir explorando
                call    Key_BCD         ;Sí. Convierte código de tecla
                                        ;a código BCD
                call    TEMPR           ;Almacena la temperatura de referencia
                movf    Tecla,0
                sublw   .9
                btfss   STATUS,0        ;¿Es mayor que 9 (A,B,C,D,E,F)?
                goto    MAYOR           ;Sí
                movf    Tecla,0         ;No
                addlw   0x30            ;Ajuste ASCII de los caracteres
                                        ;del 0 al 9
                goto    VER
MAYOR           movf    Tecla,0
                addlw   0x37            ;Ajuste ASCII de los caracteres
                                        ;de la A a la F
VER             call    LCD_DATO        ;Visualizar sobre el LCD
TECLA_P         call    Key_Scan        ;Explora el teclado
                movlw   0x80
                subwf   Tecla,1
                btfss   STATUS,2        ;¿Se ha liberado la tecla pulsada?
                goto    TECLA_P         ;No todavía
                decfsz  CONTA_C,1       ;Sí. ¿Se han visualizado 3 caracteres?
                goto    LOOP            ;No
                movlw   0xC0            ;Sí. Posiciona el cursor en
                call    LCD_REG         ;la siguiente línea

;Salida del mensaje "TIME(min):"

                movlw   MENS_1          ;Visualiza el mensaje
                call    MENSAJE
                movlw   .2              ;Inicia contador de
                movwf   CONTA_1         ;caracteres 1
                movlw   .5              ;Inicia contador de caracteres
                movwf   CUENTA          ;del tiempo
                movlw   b'00011001'     ;Constante que permite almacenar
                movwf   BASE            ;el tiempo establecido

LOOP1           call    Key_Scan        ;Explora el teclado
                movlw   0x80
                subwf   Tecla,0
                btfsc   STATUS,2        ;¿Hay alguna pulsada?
                goto    LOOP1           ;No. Seguir explorando
                call    Key_BCD         ;Sí. Convierte código de tecla
                                        ;a código BCD
                call    TIMEREF         ;Almacena el tiempo establecido
                movf    Tecla,0
                sublw   .9
                btfss   STATUS,0        ;¿Es mayor que 9 (A,B,C,D,E,F)?
```

```
                goto        MAS             ;Sí
                movf        Tecla,0         ;No
                addlw       0x30            ;Ajuste ASCII de los caracteres
                                            ;del 0 al 9
                goto        VISUAL
MAS             movf        Tecla,0
                addlw       0x37            ;Ajuste ASCII de los caracteres
                                            ;de la A a la F
VISUAL          call        LCD_DATO        ;Visualizar sobre el LCD
KEY_P           call        Key_Scan        ;Explora el teclado
                movlw       0x80
                subwf       Tecla,1
                btfss       STATUS,2        ;¿Se ha liberado la tecla pulsada?
                goto        KEY_P           ;No todavía
                decf        CUENTA,1        ;Sí
                decfsz      CONTA_1,1       ;¿Se han visualizado 2 caracteres?
                goto        LOOP1           ;No
                btfss       CUENTA,1        ;Sí. ¿Se han desplegado en el LCD los
                                            ;5 caracteres del tiempo?
                goto        PROCESO         ;Sí
                movlw       .2              ;No. Inicia contador de
                movwf       CONTA_1         ;caracteres 1
                movlw       '.'             ;Visualiza punto decimal
                call        LCD_DATO
                bsf         BASE,6
                goto        LOOP1

;*****************************************PROCESO1************************************

;PROCESO de conversión de una señal analógica en una señal digital

PROCESO         bsf         STATUS,5        ;Banco 1
                bcf         STATUS,6
                movlw       0x01            ;Habilita la interrupción del TMR1
                movwf       PIE1
                bcf         STATUS,5        ;Banco 0
                movlw       0x40            ;Habilita la interrupción de los periféricos
                movwf       INTCON          ;no controlados con INTCON
                clrf        T1CON           ;Desactiva el TMR1
                movlw       0xB0            ;Valor a partir del cual el TMR1
                movwf       TMR1L           ;empieza la temporización para
                movlw       0x3C            ;lograr una interrupción cada
                movwf       TMR1H           ;10 ms
                movlw       0x63            ;Carga el RELOJ para una temporización
                movwf       RELOJ           ;de 1 s
                clrf        MASTER          ;Inicializa registro para pasar a la etapa
                                            ;de la electroválvula
                clrf        CERO            ;Inicializa registro igual a cero
                clrf        PORTA           ;Limpia el puerto A
ADC             btfsc       MASTER,0        ;¿El bit 0 de MASTER es igual a 1?
                goto        ATLAST          ;Sí
                bsf         STATUS,5        ;No. Banco 1
                bcf         STATUS,6
                movlw       0x01            ;Configura los pines del puerto A,
                movwf       TRISA           ;RA0 entrada
                movlw       0x8E            ;Justificación a la derecha,
```

```
        movwf   ADCON1          ;RA0 entrada analógica
        bcf     STATUS,5        ;Cambio al banco 0
        movlw   0x81            ;Selecciona el reloj de conversión,
        movwf   ADCON0          ;selecciona el canal de entrada A/D,
                                ;activa el módulo A/D
        bcf     PIR1,6          ;Restaura flag del conversor
        movlw   .150            ;Temporización de 1 s aproximadamente
        movwf   ACQUIRE         ;para adquisición de datos
ACQ     call    LCD_DELAY
        decfsz  ACQUIRE,1
        goto    ACQ
        bsf     ADCON0,2        ;Inicio de la conversión
WAIT    btfss   PIR1,6          ;¿Ha terminado la conversión?
        goto    WAIT            ;No. Esperar
        bcf     PIR1,6          ;Sí. Restaura flag del conversor

;El valor binario obtenido se pasa a un valor en grados que pasa a BCD

        bsf     STATUS,5        ;Pasa al banco 1
        bcf     STATUS,6
        movf    ADRESL,0
        bcf     STATUS,5        ;Cambio al banco 0
        movwf   DIFER_L         ;Carga la parte baja del valor binario obtenido
                                ;del módulo ADC
        movf    ADRESH,0        ;Carga la parte alta del valor binario obtenido
        movwf   DIFER_H         ;del módulo ADC
        call    DIV             ;Divide el resultado de la conversión A/D por 2
        call    B_BCD           ;Convierte el valor binario presente en
                                ;COCIENT en BCD

;La temperatura sensada se visualiza en el LCD

        movf    CODIGOL,0       ;Guarda el dato para visualizar
        movwf   LAPSO           ;la unidad
        bcf     CODIGOL,0       ;Se obtiene la decena
        bcf     CODIGOL,1
        bcf     CODIGOL,2
        bcf     CODIGOL,3
        swapf   CODIGOL,1
        bcf     LAPSO,4         ;Se obtiene la unidad
        bcf     LAPSO,5
        bcf     LAPSO,6
        bcf     LAPSO,7
        movlw   0xC0            ;Posiciona el cursor al inicio de
        call    LCD_REG         ;la segunda línea
        movf    CODIGOH,0       ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza centenas de grado
        movf    CODIGOL,0       ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza decenas de grado
        movf    LAPSO,0         ;Ajuste ASCII de los caracteres
        addlw   0x30            ;del 0 al 9
        call    LCD_DATO        ;Visualiza unidades de grado
        movlw   0xDF            ;Visualiza símbolo °
        call    LCD_DATO
```

```
                movlw       0x43            ;Visualiza C
                call        LCD_DATO
                movlw       ‘ ‘
                call        LCD_DATO
                movlw       ‘ ‘
                call        LCD_DATO
                movlw       ‘ ‘
                call        LCD_DATO
                movlw       ‘ ‘
                call        LCD_DATO
```

;La temperatura sensada se compara con la temperatura de referencia

```
                movf        CODIGOH,0       ;Carga la centena de grado de la temperatura
sensada
                subwf       TRBCD0,0        ;Compara con la temperatura de referencia
                btfsc       STATUS,2        ;¿Son iguales?
                goto        IOMQUEC         ;Sí
                btfsc       STATUS,0        ;No. ¿TRBCD0 > CODIGOH?
                goto        ON_IO           ;Sí
                goto        IOMQUEU         ;No
IOMQUEC         movf        CODIGOL,0       ;Carga la decena de grado de la temperatura
                                            ;sensada
                subwf       TRBCD1,0        ;Compara con la temperatura de referencia
                btfsc       STATUS,2        ;¿Son iguales?
                goto        IOMQUED         ;Sí
                btfsc       STATUS,0        ;No. ¿TRBCD1 > CODIGOL?
                goto        ON_IO           ;Sí
                goto        IOMQUEU         ;No
IOMQUED         movf        LAPSO,0         ;Carga la unidad de grado de la temperatura
                                            ;sensada
                subwf       TRBCD2,0        ;Compara con la temperatura de referencia
                btfsc       STATUS,2        ;¿Son iguales?
                goto        IOMQUEU         ;Sí
                btfsc       STATUS,0        ;No. ¿TRBCD2 > LAPSO?
                goto        ON_IO           ;Sí
                goto        IOMQUEU         ;No
ON_IO           bsf         PORTA,2         ;Señal ON/OFF encendida
                goto        ADC             ;Realiza una nueva conversión A/D
IOMQUEU         bcf         PORTA,2         ;Apaga la señal ON/OFF
                bsf         T1CON,0         ;Activa el TMR1
                bsf         INTCON,7        ;Permiso global de interrupciones
                goto        ADC             ;Realiza una nueva conversión A/D
```

;**

;Etapa de la electroválvula

```
ATLAST          bcf         T1CON,0         ;Desactiva el TMR1
                bcf         INTCON,7        ;Permiso global de interrupciones denegado
                bcf         PORTA,2         ;Apaga la señal ON/OFF
                call        UP_LCD          ;Configuración de líneas para el LCD
                call        LCD_INI         ;Inicialización del LCD
```

;Salida del mensaje "ABRIR VALVULA"

```
                movlw   MENS_2          ;Visualiza el mensaje
                call    MENSAJE
                movlw   0xC0            ;Posiciona el cursor en
                call    LCD_REG         ;la siguiente línea

;Salida del mensaje "PRESIONAR <A>"

                movlw   MENS_3          ;Visualiza el mensaje
                call    MENSAJE
NOTYET          call    Key_Scan        ;Explora el teclado
                movlw   0x80
                subwf   Tecla,0
                btfsc   STATUS,2        ;¿Hay alguna pulsada?
                goto    NOTYET          ;No. Seguir explorando
                bsf     PORTA,1         ;Sí. Abrir electroválvula
BOTON_P         call    Key_Scan        ;Explora el teclado
                movlw   0x80
                subwf   Tecla,1
                btfss   STATUS,2        ;¿Se ha liberado la tecla pulsada?
                goto    BOTON_P         ;No todavía
                movlw   0x80            ;Sí. Posiciona el cursor en
                call    LCD_REG         ;la primera línea

;Salida del mensaje "CERRAR VALVULA"

                movlw   MENS_4          ;Visualiza el mensaje
                call    MENSAJE
                movlw   0xC0            ;Posiciona el cursor en
                call    LCD_REG         ;la siguiente línea

;Salida del mensaje "PRESIONAR <A>"

                movlw   MENS_3          ;Visualiza el mensaje
                call    MENSAJE
AUNNO           call    Key_Scan        ;Explora el teclado
                movlw   0x80
                subwf   Tecla,0
                btfsc   STATUS,2        ;¿Hay alguna pulsada?
                goto    AUNNO           ;No. Seguir explorando
                bcf     PORTA,1         ;Sí. Cerrar electroválvula
BUTTON          call    Key_Scan        ;Explora el teclado
                movlw   0x80
                subwf   Tecla,1
                btfss   STATUS,2        ;¿Se ha liberado la tecla pulsada?
                goto    BUTTON          ;No todavía

;Salida del mensaje "PRESIONAR RESET"

AVISAR          call    UP_LCD          ;Sí. Configuración de líneas para el LCD
                call    LCD_INI         ;Inicialización del LCD
                movlw   MENS_5          ;Visualiza el mensaje
                call    MENSAJE
                movlw   0xC0            ;Posiciona el cursor en
                call    LCD_REG         ;la siguiente línea

;Salida del mensaje "PARA UN NUEVO"
```

```
                movlw   MENS_6          ;Visualiza el mensaje
                call    MENSAJE
                movlw   .150            ;Temporización de 1 s aproximadamente
                movwf   ACQUIRE         ;para visualizar aviso final
AVISO1          call    LCD_DELAY
                decfsz  ACQUIRE,1
                goto    AVISO1
                call    UP_LCD          ;Configuración de líneas para el LCD
                call    LCD_INI         ;Inicialización del LCD

;Salida del mensaje "PROCESO"

                movlw   MENS_7          ;Visualiza el mensaje
                call    MENSAJE
                movlw   .150            ;Temporización de 1 s aproximadamente
                movwf   ACQUIRE         ;para visualizar aviso final
AVISO2          call    LCD_DELAY
                decfsz  ACQUIRE,1
                goto    AVISO2
                goto    AVISAR

                END                     ;Fin del programa fuente
```

```
;*******************************************************************************
;                                   TECLADO.INC
;
;Presenta un conjunto de rutinas que permiten la gestión de un teclado organizado
;en una matriz de 4 x 4.
;
;Este fichero se debe incluir en los futuros programas fuente mediante la directiva INCLUDE.
;
;Key_Scan:      Realiza un barrido del teclado y detecta si hay alguna tecla pulsada. La
;               variable 'Tecla' retorna con el código de la tecla pulsada o el 0x80 en caso
;               de no haber ninguna.
;
;Key_BCD:       Convierte el código de tecla en código BCD (del 0 al F). Antes de llamar a la
;               rutina, la variable 'Tecla' contiene el código de tecla; al finalizar, la
;               rutina devuelve el código BCD en la misma variable 'Tecla'.
;
;El teclado matricial se supone conectado al puerto B y su disposición es la siguiente:
;
;                 RB0 RB1 RB2 RB3
;                  ↑   ↑   ↑   ↑
;              +---+---+---+---+
;       RB4 →  | 1 | 2 | 3 | F |
;              +---+---+---+---+
;       RB5 →  | 4 | 5 | 6 | E |
;              +---+---+---+---+
;       RB6 →  | 7 | 8 | 9 | D |
;              +---+---+---+---+
;       RB7 →  | A | 0 | B | C |
;              +---+---+---+---+
;
;                 TECLA        CÓDIGO
;                 -----        ------
;                   0          01111101 = 0x7D
;                   1          11101110 = 0xEE
;                   2          11101101 = 0xED
;                   3          11101011 = 0xEB
;                   4          11011110 = 0xDE
;                   5          11011101 = 0xDD
;                   6          11011011 = 0xDB
;                   7          10111110 = 0xBE
;                   8          10111101 = 0xBD
;                   9          10111011 = 0xBB
;                   A          01111110 = 0x7E
;                   B          01111011 = 0x7B
;                   C          01110111 = 0x77
;                   D          10110111 = 0xB7
;                   E          11010111 = 0xD7
;                   F          11100111 = 0xE7

                CBLOCK Key_var          ;Inicio de las variables
                Tecla                   ;Retorno del código de tecla
                Key_1                   ;Nº de columnas a explorar
                Key_2                   ;Temporal de código
                Key_Delay               ;Variable de temporización
                Key_Delay_1             ;Variable de temporización
                Key_Delay_2             ;Variable de temporización
                TRISB_Temp              ;Estado temporal de TRISB
                ENDC

;*******************************************************************************
;Key_Scan: Rutina de exploración del teclado. La variable 'Tecla' retorna con el código
;de la tecla pulsada o el código 0x80 si no se pulsa ninguna.

Key_Scan        bsf     STATUS,5        ;Selecciona banco 1
                movf    TRISB,0
                bcf     STATUS,5        ;Selecciona banco 0
```

```
                movwf   TRISB_Temp      ;Salva estado actual de TRISB
                bsf     STATUS,5        ;Banco 1
                movlw   b'00001111'
                movwf   TRISB           ;RB0-RB3 entradas, RB4-RB7 salidas
                bcf     OPTION_REG,7    ;Resistencias Pull-up activadas
                bcf     STATUS,5        ;Selecciona banco 0
                movlw   .4
                movwf   Key_1           ;Nº de columnas a explorar
                movlw   b'01111111'
                movwf   Tecla           ;Fila a activar
Key_Scan_1      movf    Tecla,0
                movwf   PORTB           ;Activa fila
                nop                     ;Explorar columnas
                nop
                nop
                movf    PORTB,0
                movwf   Key_2
                subwf   Tecla,0
                btfss   STATUS,2        ;¿Hay alguna pulsada?
                goto    Key_Scan_2      ;Si hay alguna pulsada
                bsf     STATUS,0        ;No hay ninguna en esa fila
                rrf     Tecla,1         ;Selecciona siguiente fila
                decfsz  Key_1,1
                goto    Key_Scan_1
                movlw   0x80
                movwf   Tecla           ;Retorna código 0x80 (no hay pulsación)
                movf    TRISB_Temp,0
                bsf     STATUS,5        ;Selecciona banco 1
                movwf   TRISB           ;Repone TRISB al valor original
                bsf     OPTION_REG,7    ;Resistencias Pull-up desactivadas
                bcf     STATUS,5        ;Selecciona banco 0
                return                  ;Fin de exploración

Key_Scan_2      movlw   .255            ;Bucle de temporización de unos 20 ms
                movwf   Key_Delay       ;para evitar el rebote de los pulsado-
                movlw   .138            ;res
                movwf   Key_Delay_1
Key_Scan_3      clrf    Key_Delay_2
                clrwdt
Key_Scan_4      decfsz  Key_Delay_2,1
                goto    Key_Scan_4
                decfsz  Key_Delay,1
                goto    Key_Scan_3
                movlw   .1
                movwf   Key_Delay
                decfsz  Key_Delay_1,1
                goto    Key_Scan_3

                movf    Tecla,0         ;Tras la temporización se lee nuevamente
                movwf   PORTB           ;si la tecla es la misma. Así se evitan
                nop                     ;los rebotes
                nop
                nop
                movf    PORTB,0
                subwf   Key_2,0
                btfss   STATUS,2        ;¿Es la misma?
```

```
                goto    Key_Scan_1          ;No, seguir con la exploración
                movf    Key_2,0             ;Sí, guardar en variable de salida 'Tecla'
                movwf   Tecla
                movf    TRISB_Temp,0
                bsf     STATUS,5            ;Selecciona banco 1
                movwf   TRISB               ;Repone TRISB al valor original
                bsf     OPTION_REG,7        ;Resistencias Pull-up desactivadas
                bcf     STATUS,5            ;Selecciona banco 0
                return                      ;Fin de exploración

;*******************************************************************************
;Key_BCD: Convierte el código de tecla que haya en la variable 'Tecla' a BCD (0 - F). El
;resultado se devuelve en la misma variable 'Tecla'.

Key_Tabla       movf    Key_1,0
                addwf   PCL,1               ;Calcula desplazamiento
                retlw   b'01111101'         ;0
                retlw   b'11101110'         ;1
                retlw   b'11101101'         ;2
                retlw   b'11101011'         ;3
                retlw   b'11011110'         ;4
                retlw   b'11011101'         ;5
                retlw   b'11011011'         ;6
                retlw   b'10111110'         ;7
                retlw   b'10111101'         ;8
                retlw   b'10111011'         ;9
                retlw   b'01111110'         ;A
                retlw   b'01111011'         ;B
                retlw   b'01110111'         ;C
                retlw   b'10110111'         ;D
                retlw   b'11010111'         ;E
                retlw   b'11100111'         ;F

Key_BCD         movf    Tecla,0
                movwf   Key_2               ;Almacena el código temporalmente
                clrf    Key_1               ;Contador BCD a 0
Key_BCD_2       call    Key_Tabla           ;Busca código en la tabla
                subwf   Key_2,0             ;Compara con el de la tecla
                btfsc   STATUS,2            ;¿Coincide?
                goto    Key_BCD_1           ;Sí
                incf    Key_1,1             ;No, incrementa contador BCD
                goto    Key_BCD_2
Key_BCD_1       movf    Key_1,0
                movwf   Tecla               ;Carga contador BCD en la variable de salida
                return

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

```
;                                       LCD_CXX.INC
;
#define ENABLE          bsf PORTE,2     ;Activa señal E
#define DISABLE         bcf PORTE,2     ;Desactiva señal E
#define LEER            bsf PORTE,1     ;Pone LCD en Modo RD
#define ESCRIBIR        bcf PORTE,1     ;Pone LCD en Modo WR
#define OFF_COMANDO     bcf PORTE,0     ;Desactiva RS (modo comando)
#define ON_COMANDO      bsf PORTE,0     ;Activa RS (modo dato)

            CBLOCK      Lcd_var         ;Inicio de las variables. Será la primera
            LCD_Temp_1                  ;dirección libre disponible
            LCD_Temp_2
            ENDC

;*****************************************************************************************
;UP_LCD: Configuración PIC para el LCD.

UP_LCD      clrf        PORTB           ;Limpia salidas
            clrf        PORTE
            bsf         STATUS,5        ;Banco 1
            movlw       b'10001110'     ;Configura todos los pines del
            movwf       ADCON1          ;puerto E como E/S digitales
            clrf        TRISB           ;RB <0-7> salidas digitales
            movlw       b'00000000'
            movwf       TRISE           ;RE0-RE2 salidas
            bcf         STATUS,5        ;Banco 0
            OFF_COMANDO                 ;RS=0
            DISABLE                     ;E=0
            return

;*****************************************************************************************
;LCD_BUSY: Lectura del Flag Busy y la dirección.

LCD_BUSY    LEER                        ;Pone el LCD en Modo RD
            bsf         STATUS,5        ;Banco 1
            movlw       b'11111111'
            movwf       TRISB           ;Puerto B como entrada
            bcf         STATUS,5        ;Selecciona el banco 0
            ENABLE                      ;Activa el LCD
            nop
            nop
            nop
LCD_BUSY_1  btfsc       PORTB,7         ;Chequea bit de Busy
            goto        LCD_BUSY_1
            DISABLE                     ;Desactiva LCD
            bsf         STATUS,5        ;Banco 1
            clrf        TRISB           ;Puerto B salida
            bcf         STATUS,5
            ESCRIBIR                    ;Pone LCD en modo WR
            return

;*****************************************************************************************
;LCD_E: Pulso de Enable.

LCD_E       ENABLE                      ;Activa E
```

```
                nop
                nop
                nop
                DISABLE                         ;Desactiva E
                return

;*****************************************************************************
;LCD_DATO: Escritura de datos en DDRAM o CGRAM. Envía el dato presente en el W.

LCD_DATO        OFF_COMANDO                     ;Desactiva RS (modo comando)
                movwf       PORTB               ;Valor ASCII a sacar por PORTB
                call        LCD_BUSY            ;Espera a que se libere el LCD
                ON_COMANDO                      ;Activa RS (modo dato)
                goto        LCD_E               ;Genera pulso de E

;*****************************************************************************
;LCD_REG: Escritura de comandos en el LCD. Envía el comando presente en el W.

LCD_REG         OFF_COMANDO                     ;Desactiva RS (modo comando)
                movwf       PORTB               ;Código de comando
                call        LCD_BUSY            ;¿LCD libre?
                goto        LCD_E               ;Sí. Genera pulso de E

;*****************************************************************************
;LCD_INI: Inicialización del LCD enviando el comando 'Function Set' 3 veces consecutivas
;con un intervalo de unos 5 ms.

LCD_INI         movlw       b'00111000'
                call        LCD_REG             ;Código de instrucción
                call        LCD_DELAY           ;Temporiza
                movlw       b'00111000'
                call        LCD_REG             ;Código de instrucción
                call        LCD_DELAY           ;Temporiza
                movlw       b'00111000'
                call        LCD_REG             ;Código de instrucción
                call        LCD_DELAY           ;Temporiza
                movlw       b'00001111'         ;LCD On, Cursor On y Blink On
                call        LCD_REG
                movlw       b'00000001'         ;Borra LCD y Home
                call        LCD_REG
                movlw       b'00000110'         ;Cursor incrementa su posición
                call        LCD_REG             ;después de cada caracter
                return

;*****************************************************************************
;LCD_DELAY: Rutina de temporización de unos 5 ms. Se emplean las variables LCD_Temp_1
;y LCD_Temp_2.

LCD_DELAY       clrwdt
                movlw       .100
                movwf       LCD_Temp_1
                clrf        LCD_Temp_2
LCD_DELAY_1     decfsz      LCD_Temp_2,1
                goto        LCD_DELAY_1
                decfsz      LCD_Temp_1,1
                goto        LCD_DELAY_1
```

```
                return

;*******************************************************************************************
;CHECARE: Rutina que reconfigura el LCD para visualizar la centésima de segundo del
;tiempo establecido en la posición CE del LCD.

CHECARE         movlw           0xCE                    ;Coloca el cursor en la posición de
                call            LCD_REG                 ;la centésima de segundo
                return

;*******************************************************************************************
;CHECARD: Rutina que reconfigura el LCD para visualizar la décima de segundo del
;tiempo establecido en la posición CD del LCD.

CHECARD         movlw           0xCD                    ;Coloca el cursor en la posición de
                call            LCD_REG                 ;la décima de segundo
                return

;*******************************************************************************************
;CHECARB: Rutina que reconfigura el LCD para visualizar la unidad de minuto del
;tiempo establecido en la posición CB del LCD.

CHECARB         movlw           0xCB                    ;Coloca el cursor en la posición de
                call            LCD_REG                 ;la unidad de minuto
                return

;*******************************************************************************************
;CHECARA: Rutina que reconfigura el LCD para visualizar la decena de minuto del
;tiempo establecido en la posición CA del LCD.

CHECARA         movlw           0xCA                    ;Coloca el cursor en la posición de
                call            LCD_REG                 ;la decena de minuto
                return
```

IX.2 SISTEMA DE CONTROL DE TEMPERATURA

Figura 36. Vista exterior del control de temperatura desarrollado.

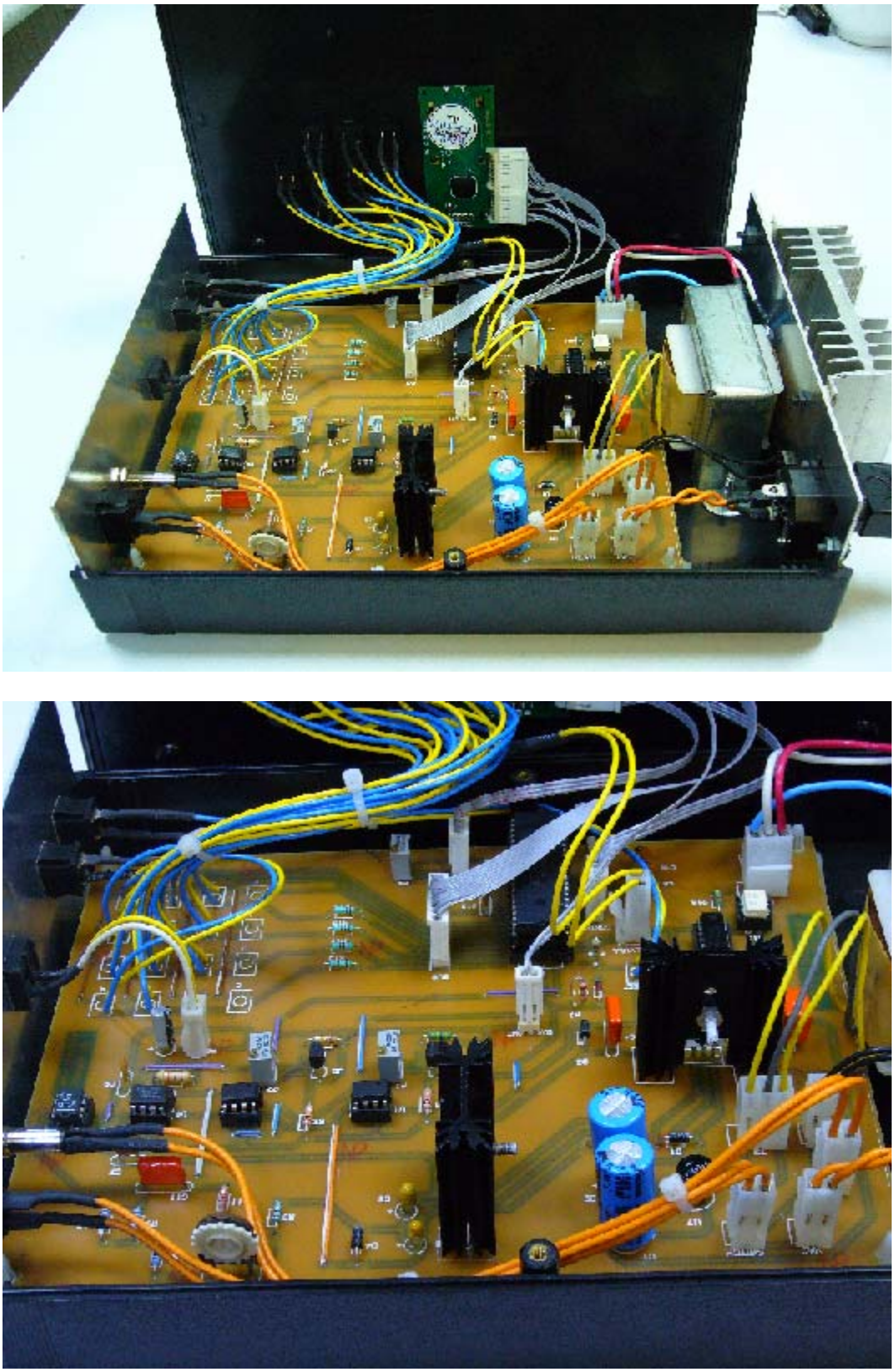

Figura 37. Vista interior y circuito impreso del control de temperatura desarrollado.

IX.3 CIRCUITO IMPRESO

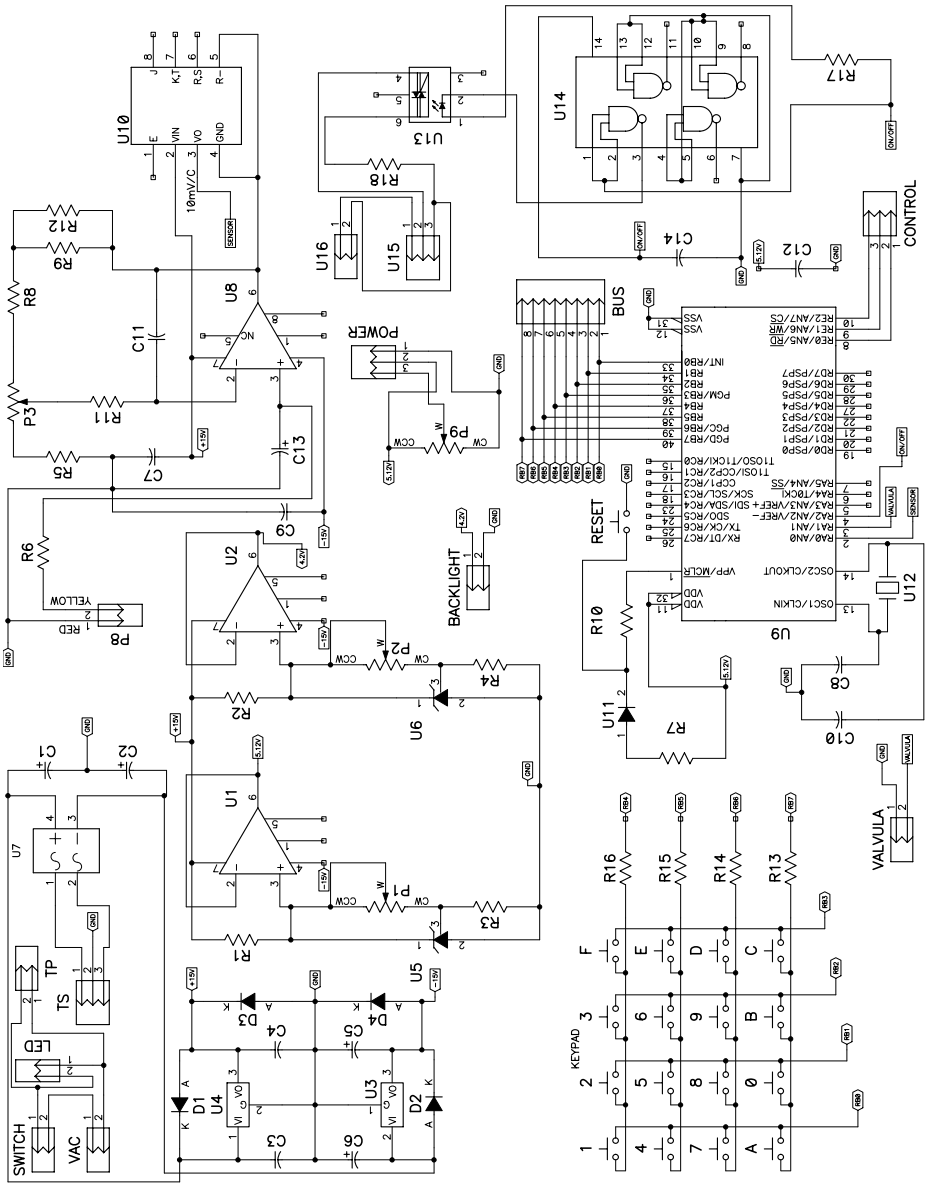

Figura 38. Esquemático.

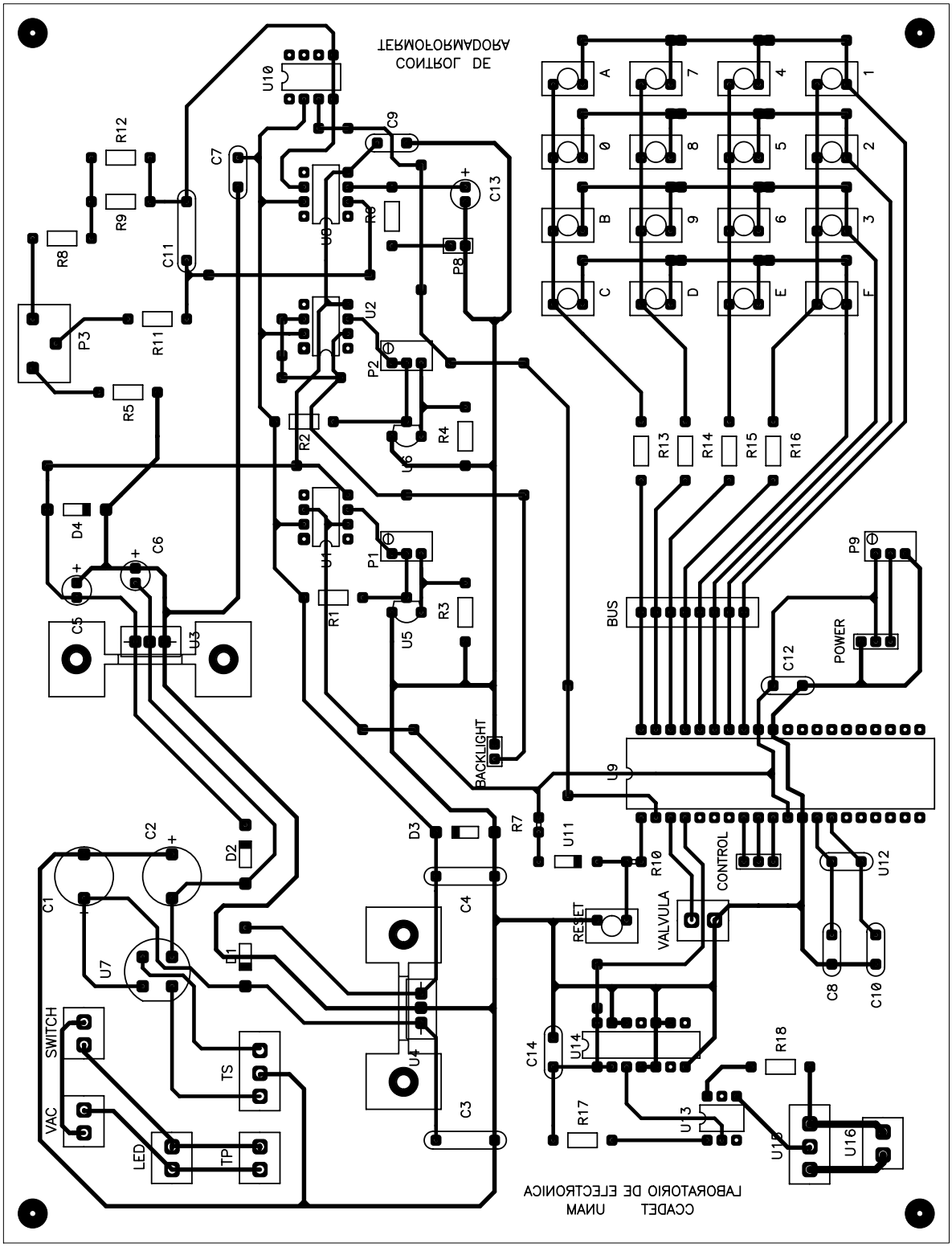

Figura 39. PCB.