UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

Instituto de Ciencias Básicas e Ingeniería

Control de Temperatura por Medio de un Autómata Basado en el Microcontrolador PIC16F877.

T E S I S

que para obtener el título en

Ingeniería en Electrónica y Telecomunicaciones

P R E S E N T A N

Verónica Gómez Mejía

Pedro Alfredo Cerón Torres

Asesores: Dr. Virgilio López Morales

Ing. Julio Cesar Ramos Fernández[1]

PACHUCA DE SOTO, HIDALGO. OCTUBRE DE 2006

[1] Investigador de la Universidad Tecnológica de Tula-Tepeji

Agradecimientos

A mis tíos: Thomas, León, Fortunato, Luy y Angeles, por su motivación y apoyo sin el cual ningún logro sería alcanzable.

A mis primos: Leoncito, Sinuhé, José, Miguel y Cesar, por ser como mis hermanos.

A mis amigos inolvidables: Mayen, Areli, Dulce, Alfredito, Nahum, Victor, Nepe, Gildardo, Jahen, Marco, por el cariño que me han brindado.

Vero.

A mis padres: Raymunda Torres Cruz y Margarito Cerón García, por su amor, su confianza y comprensión.

A Verito y a mis amigos, ya que sin ellos habría sido más difícil terminar.

Cerón.

A nuestros asesores, Virgilio López Morales y Julio Cesar Ramos Fernández, por su apoyo para concluir esta tesis y a cada uno de los profesores que a lo largo de la carrera nos enseñaron algo.

Índice general

Índice de figuras

Índice de tablas

Resumen

El proyecto consiste en el diseño y simulación, tanto digital como analógica, de un algoritmo basado en autómatas finitos deterministas, con el fin de obtener dos señales de salida, que pueden ser usadas para controlar la apertura y cierre de las cortinas de ventilación de un invernadero por niveles, de acuerdo a las condiciones del clima externo. Para el diseño del algoritmo se consideran tres variables meteorológicas: temperatura exterior, velocidad de viento y presencia de lluvia.

Para proponer los valores para los cuales se deben abrir o cerrar la cortinas, se toma en cuenta que, para evitar daños en la plantación, se deben mantener cerradas mientras exista lluvia, la temperatura exterior sea muy baja o, la velocidad del viento sea muy alta; para cuando no exista lluvia, la temperatura exterior aumente o, la velocidad del viento disminuya, las cortinas se mantendrán en un nivel de apertura del 33 %, 66 % ó 100 %.

Para el control del sistema de las cortinas se diseña un autómata que, de acuerdo a los valores propuestos para cada una de las variables meteorológicas, debe abrir o cerrar las cortinas hasta el nivel correspondiente. Este autómata se adapta primero para ser programado en MATLAB y realizar la simulación con los valores propuestos.

El autómata se debe adaptar también para programar el algoritmo en un sistema mínimo basado en el microcontrolador PIC16F877 de Microchip. Este microcontrolador cuenta con un convertidor analógico a digital con entradas analógicas multiplexadas en las que se conectan divisores de tensión con potenciómetros, los cuales simulan las señales de temperatura exterior y velocidad del viento.

Para simular la presencia de lluvia y la posición de las cortinas de ventilación se utilizan interruptores normalmente abiertos, conectados a entradas digitales del microcontrolador. Una vez que se cuenta con el valor de cada variable, se obtienen las señales de salida las cuales controlan la apertura y el cierre de las cortinas de ventilación.

Para visualizar el valor de cada variable y de las señales de salida se utiliza una pantalla de cristal líquido en el sistema mínimo. También se hace uso de LabVIEW para llevar a cabo la instrumentación virtual del sistema desde una computadora utilizando la interfaz de comunicación serial utilizando el protocolo RS-232.

Objetivo general

Diseñar y programar un algoritmo de control basado en un autómata finito determinista para mejorar la temperatura interna de un invernadero dentro de un rango apropiado para el cultivo, aprovechando las perturbaciones del clima externo.

Para cumplir con lo anterior se plantean los siguientes:

Objetivos particulares

- Conocer las principales variables de acuerdo a su impacto sobre el cultivo y su relación con la ventilación.
- Proponer los rangos de las variables para la apertura y cierre de las cortinas.
- Diseñar un autómata que evalúe los estados del sistema y de como resultado una acción de control para la apertura y cierre de las cortinas.
- Programar un algoritmo de control para el microcontrolador, basado en el autómata diseñado.
- Supervisar el comportamiento del autómata mediante LabVIEW utilizando la interfaz serial RS-232.

Justificación

Cuando se requiere regular la temperatura interior de un invernadero, es común utilizar sistemas de calefacción y de ventilación. Una forma de disminuir o regular la temperatura interior es el empleo de cortinas de ventilación. Este sistema proporciona grandes ventajas ya que se puede mantener la temperatura sin utilizar sistemas de refrigeración o aire forzado, los cuales consumirían más energía, o bien, en condiciones extremas mantener la temperatura con la ayuda de dichos sistemas.

Para controlar la apertura y cierre de las cortinas de ventilación se propone la implementación en un sistema mínimo de un algoritmo basado en un autómata finito determinista, lo que significa una disminución en los gastos de implementación, debido a que con un dispositivo de reducidas dimensiones y pocos componentes, se lleva a cabo la adquisición de señales analógicas y digitales para así obtener como salida una señal de control.

Además, otra ventaja es la posibilidad de supervisar las variables meteorológicas y el estado del autómata, al interconectar el sistema mínimo y una computadora personal mediante el protocolo de comunicación serial RS-232, lo que permite la instrumentación virtual del autómata.

El autómata también se puede utilizar para la simulación y entrenamiento cuando se desee conocer el comportamiento del sistema de cortinas del invernadero al cambiar las condiciones meteorológicas externas

Capítulo 1

Introducción

1.1. El invernadero

Un invernadero es un recinto cerrado donde la diferencia entre el clima interior y exterior se crea principalmente mediante la radiación solar y los mecanismos del estancamiento del aire. Todo esto es con el objetivo de proveer un ambiente apropiado para el cultivo de especies vegetales, [1]. Algunas de las ventajas de cultivar en invernaderos pueden ser las siguientes:

- Provee un microclima especial para el mejor crecimiento de los cultivos.
- Logra extender los tiempos de producción.
- Protege a los cultivos de las inclemencias del tiempo y de plagas.

Hay un axioma que es digno de tenerse en cuenta: *quien menos produce incurre en más costos, porque la productividad es inversamente proporcional a los requerimientos de agroquímicos*, [1].

A menor productividad mayor uso de agroquímicos. Cuando una planta no es productiva es porque ha tenido problemas de exceso o falta de humedad, de exceso o falta de temperatura, de exceso o falta de ventilación, de exceso o falta de luminosidad, es decir ha tenido problemas derivados de un mal diseño y mal manejo en la plantación, [1].

1.1.1. Factores a considerar para el funcionamiento del invernadero

Los invernaderos no pueden ser copias improvisadas de otros invernaderos, sino el resultado de un estudio puntual del clima existente, del microclima que debe proveerse a la plantación programada y de los diversos factores que intervienen en diferentes

épocas del año en la zona en la que se implantará el mismo.

Aun en los casos en los que la construcción se haga en las mismas zonas, no se deben hacer copias exactas o clones ya que pueden requerirse variaciones importantes en la estructura para conseguir los resultados deseados, [2], [3].

Es importante tener en cuenta que la altura, las ventanas cenitales y el ancho de las naves juega un papel decisivo, por lo tanto todo nuevo invernadero debe ser objeto de un estudio particular de ambientación climática.

Son muchos los factores a considerar para la construcción de invernaderos, dado que si existen errores en el diseño es muy probable, por ejemplo, obtener niveles de humedad relativa muy altos, lo cual contribuye al desarrollo de plagas y enfermedades, e induce a aplicar agroquímicos de elevados costos reduciendo la calidad de los frutos. Consecuentemente el productor pierde gran parte de la cosecha debido a que la plantación no se transforma en frutos o crece con malformaciones, [2].

En un invernadero de ambientación climática natural, el único motor que cumple la función de regulador de temperaturas y humedad relativa es el viento, pues expulsa los excedentes de temperatura, lo cual es necesario para que se lleve a cabo la polinización. Debido a esto, en el diseño se debe considerar la altura del invernadero y las dimensiones de las aperturas cenitales para que exista, en ese espacio, el volumen de aire requerido y se produzca la renovación necesaria, [2].

Por ello el viento es una de las variables más importante para obtener buenos resultados y, esta variable puede señalar la manera precisa para construcción así como las condiciones apropiadas para poder ejecutar de manera profesional las técnicas de manejo, [2].

Esto explica porque es tan necesario que un invernadero además de ser un área protegida también deba ser controlada y establecida para evitar que la plantación se exponga a todos los factores que pudieran perjudicar sus resultados. Algunos de estos factores se describen enseguida.

Exceso de humedad relativa

La humedad atmosférica de un invernadero interviene en la transpiración, en el crecimiento de los tejidos, en la fecundación de las flores y también en el desarrollo de enfermedades criptogámicas. Su influencia es tal, que por ejemplo, con escasez de humedad en el ambiente, la planta puede deshidratarse paralizando su desarrollo ya que durante el día, a medida que aumenta la temperatura del aire baja la humedad relativa

en el interior del invernadero, [4].

De no ser controlada la ventilación desde el diseño, el área queda muy vulnerable a que se incremente la humedad relativa y, por tanto a que se desarrollen plagas y enfermedades que pondrían en peligro la producción e incrementarían de modo sustantivo los costos de operación por la aplicación de agroquímicos para enfrentarlas.

El exceso de humedad relativa (mayor al 90 %), contribuye al crecimiento de esporas de algunos organismos patógenos, lo que quiere decir que si una plantación es controlada eficientemente para que la humedad ambiente esté por debajo de este porcentaje el éxito está prácticamente asegurado, [5].

En sentido contrario un diseño que no considere y resuelva este aspecto o un descuido en el manejo sería notablemente costoso para el productor tanto en la reducción de la productividad como en la calidad de los frutos y en el incremento de los costos de producción.

Contenido de CO_2

El bióxido de carbono(CO_2), forma parte de la composición del aire y es imprescindible para la fotosíntesis. Se puede considerar como promedio, que la mayoría de las especies tiene su óptimo de fotosíntesis entre 600 y 900 ppm (partes por millón) de CO_2, [4].

En el interior del invernadero a veces resulta difícil la renovación del aire, por eso cuando se produce la fotosíntesis durante el día, suele haber menos concentración de CO_2 que en el exterior, pero durante la noche, debido a la respiración de las plantas las proporciones se invierten.

Todos estos factores están afectados por la cubierta del invernadero y la principal razón es que la cubierta encierra el aire, lo cual induce una considerable reducción en el intercambio de aire entre el ambiente sembrado y la atmósfera exterior además de una fuerte disminución de la velocidad local del aire. Por lo tanto, ni la energía absorbida ni el agua transpirada será fácilmente liberada al exterior, además, se altera el intercambio de CO_2 con la atmósfera.

Temperatura

La temperatura es un factor muy importante, ya que en general el crecimiento y/o desarrollo de los cultivos se detiene por debajo de los 10-12^{o}C y por encima de los 30-32^{o}C. También la fotosíntesis se puede ver perjudicada o beneficiada según los rangos

de temperatura que se suministren al área controlada y su aproximación o diferencia con las temperaturas óptimas para el desarrollo adecuado de las plantas protegidas, [4].

El control de las temperaturas, aprovechando la velocidad y dirección del viento, contribuye a aumentar la productividad, mejorar la calidad de los frutos y reducir los riesgos y costos derivados de la utilización de agroquímicos.

Lluvias

Los potenciales ingresos de gotas o lluvia impulsada por el viento, deben ser controlados desde el diseño, ya que estos factores aumentan la humedad en el área con las consecuencias referidas en el párrafo referente a exceso de humedad relativa.

Viento

El intercambio de aire entre el interior y exterior del invernadero incide de una manera clara en el clima del cultivo. No sólo cambia el balance energía y la temperatura del aire, sino que también afecta al contenido de vapor de agua y de CO_2, [4]. La ventilación pasiva o natural tiene su origen en dos factores:

- Distribución de las presiones en la superficie de la estructura debido al viento.
- Diferencia de temperatura, y por lo tanto de presión, entre el invernadero y el exterior.

En el caso de invernaderos sencillos, el balance térmico se logra aprovechando la velocidad y la dirección de los vientos, [6]. En ambientación térmica natural, el viento ejerce el papel de motor del invernadero y contribuye eficazmente a:

- Balancear las temperaturas.
- Reducir la humedad relativa.
- Polinizar las plantas.
- Oxigenar la plantación.

La dirección del viento es determinante en función de varios aspectos en cuanto a la construcción del invernadero, por ejemplo:

- Protección de la estructura porque esta debe situarse en dirección que evite ser dañada por vientos extremos.
- Evitar que los gases acumulados en la parte superior circulen entre las plantas.
- La orientación del cultivo para la ventilación de las plantas y su consiguiente polinización.

Radiación térmica

Entre el 60 y 70 % de las pérdidas energéticas se originan a través del techo del invernadero. La reducción de estas pérdidas se hace a través de la selección de materiales de cobertura y pantalla adecuados como del correcto aislamiento, [4]. El uso de pantallas o mallas de aislamiento tiene varias cualidades, entre ellas:

- Durante el día la malla reduce los aportes de energía radiante. Idealmente, ésta debe ser transparente a la radiación solar visible (parte fotosintéticamente activa) y reflectante y/o absorbente a la radiación infrarroja de origen solar. Esto que permite simultáneamente producir un sombreado y evitar elevaciones de temperaturas demasiado fuertes.
- Durante la noche su función es reducir las pérdidas de energía actuando como una pantalla térmica a las radiaciones infrarrojas, además de presentarse como una resistencia complementaria a las pérdidas de energía por conducción y convección.

El sistema de sombreo con mallas aislantes resulta interesante y útil en estaciones calurosas, aunque no brinda la posibilidad de graduación o control, ya que sombrean de manera constante y reducen tanto la transmisión de la radiación fotoactiva como la del infrarrojo corto.

La gama de mallas con distinto porcentaje de transmisión, reflexión y porosidad del aire es muy amplia. Existen también materiales aluminizados que presentan la ventaja de reflejar parte de la radiación solar en lugar de absorberla.

Aves e insectos

Dentro de un invernadero deben brindarse condiciones apropiadas para que el cultivo esté a salvo de pájaros y otros animales que pudieran perjudicar los resultados de la plantación y/o traer consigo plagas y enfermedades.

Desde el diseño se pueden prever los potenciales insectos que pudieran amenazar una plantación y crear sistemas y métodos para prevenirlos, reducirlos o controlarlos. Este control, dependiendo del grado de incidencia, del diseño y del manejo de la temperatura, la humedad y el viento, pudiera hacerse sin agroquímicos con controles orgánicos o biológicos, contribuyendo a mejorar la calidad de los frutos y a reducir el impacto ambiental.

1.2. Invernaderos automatizados

La tecnología de invernaderos permite, usando recursos naturales tales como la energía solar proteger las plantas de forma que éstas sobrevivan en condiciones óptimas. Así,

las necesidades de mano de obra se reducen a las labores propias de los cultivos que se llevan a cabo en el interior de los mismos, [7].

Mucho se espera que la automatización agrícola continúe en este nuevo milenio. Pueden existir varias razones por las qué prototipos automatizados desarrollados en el siglo XX todavía no han llegado a comercializarse y la investigación se enfocará a investigar estas razones y superar estas barreras en un futuro.

Mientras estos resultados de la investigación no se pueden predecir exactamente, puede pronosticarse razonablemente que los esfuerzos en el siglo XXI serán dirigidos hacia la superación de algunas barreras referentes al progreso en tecnología de las telecomunicaciones que a la vez pueda tener un impacto importante sobre el desarrollo de la automatización agrícola en el nuevo siglo.

El ancho de banda disponible en las telecomunicaciones a un precio accesible podría aumentar considerablemente su demanda en años venideros. Esto permitirá que las imágenes de vídeo sean transmitidas económicamente y en tiempo real desde un vehículo agrícola inalámbrico a una estación fija o a alguna otra parte, y así poder supervisar remotamente la operación de tales vehículos. Así junto con otras tecnologías, este desarrollo probablemente permitirá que en el siglo XXI aparezcan robots capaces de realizar una variedad de tareas muy precisas en la agricultura.

Un ejemplo sería un sistema de sensores que permitieran reconocer la madurez de plantas confiablemente, así los robots serían capaces de cosechar las frutas y vegetales individualmente según su tamaño y madurez. Esto eliminaría la necesidad posterior de volver a seleccionar la cosecha, permitiendo que los alimentos fueran empaquetados inmediatamente después de ser recolectados, [8].

Las tentativas de utilizar técnicas modernas del control para mejorar la calidad del control en invernaderos, hasta la fecha, no han sido completamente satisfactorias debido a la complejidad de las mismas. Como ya se mencionó, se espera que estas técnicas conduzcan a obtener mejoras en un futuro no tan distante. Las nuevas y poco convencionales técnicas basadas en el cómputo inteligente han empezado a tomar finalmente su cauce en la agricultura. Algunos sistemas inteligentes ya están ganando rápidamente la aceptación. Aunque nadie puede afirmar que el control inteligente sea una panacea para cada problema concebible en la agricultura, esto debe ser considerado seriamente en un desarrollo futuro, [9].

La evolución relativa de sistemas de computo ha estallado en años recientes como resultado de la disponibilidad de plataformas de cómputo de gran alcance y a bajo costo.

En fechas recientes algunos investigadores se han enfocado a resolver problemas es-

pecíficos en la agricultura, entre ellos:

Van Straten analiza el uso de métodos de control óptimos para el cultivo en el invernadero. La penetración práctica de estos métodos sigue siendo absolutamente modesta debido a una carencia de ventajas obtenidas de pruebas experimentales con las cuales se evaluaron los riesgos implicados, así como ciertas limitaciones teóricas. La propuesta alternativa, sería reducir los costos operacionales sin depender de modelos de cultivo y con una penetración gradual de un sistema de toma de decisiones permitiendo así el aumento del crecimiento y del desarrollo a largo plazo de los sistemas de cultivo, [10].

Esta propuesta es seguida por Young quien presenta algunos progresos recientes en el modelado y control del clima y la ventilación de invernaderos. Los autores proponen una innovadora base de datos usando datos experimentales para derivar modelos dinámicos de un invernadero sin previas suposiciones de la naturaleza física del sistema. El modelo explica no sólo los datos observados, sino que busca también una forma de poder proporcionar las bases para el diseño de los sistemas de control avanzados optimizados en términos de funciones lineales cuadráticas y multiobjectivas, [11].

Linker propone el uso de modelados híbridos en los cuales redes neuronales artificiales que usan funciones radiales se utilizan para aprender a identificar fallas en un invernadero. La detección temprana de mal funcionamiento en un invernadero es claramente significativo cuando la productividad y la calidad del producto no pueden ser comprometidos.

Sigrimis propone una nueva técnica para el ahorro de la energía en invernaderos que requiere de un conocimiento limitado sobre la planta específica. El sistema propuesto se puede modificar para adaptarse fácilmente por el usuario en diferentes situaciones climáticas y hacer un promedio de períodos que puedan ser analizados e investigados, [9].

Así también, Gieling discute el problema de controlar la irrigación de agua para balancear la demanda de transpiración. La demanda del agua se trata como disturbio en un sistema la regeneración diseñado mediante un feedback convencional de tres condiciones, [12].

Sin embargo, sistemas de riego localizado en el interior de los invernaderos suponen un enorme ahorro de agua, lo cual se hará más evidente en pocas décadas, cuando la carencia de este vital líquido sea más notable, esto es debido a que se anulan prácticamente los efectos del viento sobre la evapotranspiración del cultivo. También cabe mencionar la posibilidad de realizar tratamientos fitosanitarios más efectivos con el consecuente ahorro por este concepto, [7].

El control de clima computarizado en invernaderos suele emplearse para controlar la

calefacción y ventilación, para suministrar agua y diluir nutrientes imprescindibles, [12].

Con su automatización se logra mantener dentro del rango deseable a las variables, ya que se usan técnicas de predicción o anticipación de las situaciones futuras, con el fin de asegurar el buen funcionamiento de los sistemas de producción, [7], [3].

1.3. Sistemas de control automático

El control automático ha jugado un papel importante en el avance de la ingeniería y de la ciencia. Los avances en la teoría y práctica del control automático brindan medios para lograr el funcionamiento óptimo de los sistemas dinámicos, mejorar la calidad y abaratar los costos de producción, expandir el ritmo de producción, liberar de la complejidad de muchas rutinas, de las tareas manuales repetitivas, [13].

Un sistema es una combinación de componentes que actúan conjuntamente y cumplen un determinado objetivo. Un sistema no esta limitado a los objetivos físicos. Este concepto puede ser aplicado a fenómenos abstractos y dinámicos.

La palabra *automático* implica que existe cierta sofisticación en el sistema de control. Por lo general automático significa que el sistema es normalmente capaz de adaptarse a una gama variable de condiciones de operación y que es capaz de responder a cierta variedad de entradas de manera satisfactoria. Sin embargo no todos los sistemas de control tienen la característica de ser automáticos, normalmente, esta característica se consigue realimentando la variable de salida y comparándola con la señal de mando, [14].

Es importante mencionar que el elemento más importante de cualquier sistema de control automático es el lazo de control realimentado básico, [15]. En un lazo de control realimentado la señal controlada debe realimentarse y compararse con la entrada de referencia y debe enviarse una señal de acción proporcional a la diferencia de la salida y de la entrada a través del sistema para corregir el error, [14].

1.3.1. Clasificación de los sistemas de control

Los sistemas de control se clasifican en sistemas de lazo abierto y sistemas de lazo cerrado, [13], [15], [16]. La distinción la determina la acción de control, que es la que activa al sistema para producir la salida.

Sistema de control de lazo abierto

Un sistema de control de lazo abierto es aquel en el cual la acción de control es independiente de la salida.

Los sistemas de control a lazo abierto tienen dos rasgos sobresalientes.

- La habilidad que éstos tienen para ejecutar una acción con exactitud está determinada por su calibración. Calibrar significa establecer o restablecer una relación entre la entrada y la salida con el fin de obtener del sistema la exactitud deseada.
- Estos sistemas no tienen el problema de la inestabilidad, que presentan los de lazo cerrado.

Sistema de control de lazo cerrado

Un sistema de control de lazo cerrado es aquel en el cual la acción de control es dependiente de la salida. Los sistemas de control de lazo cerrado se llaman comúnmente sistemas de control por realimentación (o retroacción).

Características de los sistemas lazo cerrado o de realimentación.

- Aumento de la exactitud. Por ejemplo, la habilidad para reproducir la entrada fielmente.
- Reducción de la sensibilidad de la salida, correspondiente a una determinada entrada, ante variaciones en las características del sistema.
- Efectos reducidos de la no linealidad. Cuando la entrada y la salida coinciden, no se requiere acción de control. Cuando existe una diferencia entre ambas, el dispositivo de comparación suministra una señal de acción de control al controlador.

La realimentación es la propiedad de un sistema de lazo cerrado que permite que la salida (o cualquier otra variable controlada del sistema) sea comparada con la entrada al sistema (o con una entrada a cualquier componente interno del mismo con un subsistema) de manera tal que se pueda establecer una acción de control apropiada como función de la diferencia entre la entrada y la salida.

La realimentación se puede efectuar por medio de una conexión eléctrica o mecánica que vaya desde los instrumentos de navegación que miden la dirección hasta el dispositivo de comparación.

1.4. Adquisición de señales

Los sistemas de adquisición se utilizan para medir y registrar señales obtenidas de dos maneras, [17].

- Aquellas originadas a partir de la medición directa de cantidades eléctricas, como pueden ser voltajes de CA o CD, frecuencia o resistencia; suelen hallarse en las áreas de prueba de componentes electrónicos, estudios ambientales y trabajos de control de calidad.
- Señales que se originan a partir de transductores, como galgas extensiométricas y termopares.

Los sistemas de adquisición se pueden clasificar en dos clases principales: analógicos y digitales. Los sistemas analógicos tratan en forma analógica la información de las mediciones. Un sistema analógico se puede definir como una función continua, como una gráfica de voltaje contra tiempo, [17], [18].

Los sistemas digitales manejan la información en forma digital. Una cantidad digital puede consistir en un número de pulsos discretos y discontinuos cuya relación de tiempo contiene información referente a la magnitud o naturaleza de la cantidad.

1.4.1. Sistemas digitales de adquisición de señales

Un sistema digital de adquisición de señales puede incluir alguno o todos los elementos de la Figura 1.1. las operaciones esenciales dentro de un sistema digital incluyen: manipulación de señales analógicas, medición, conversión y manejo de datos digitales, y programación y control interno, [17]. A continuación se describe la función de cada elemento del sistema de la Figura 1.1.

Figura 1.1: Sistema de adquisición de datos digital.

Transductor

Transforma parámetros físicos en señales eléctricas aceptables para el sistema de adquisición. Algunos parámetros son temperatura, velocidad, aceleración y presión. También es posible medir directamente cantidades eléctricas, como voltaje, resistencia o frecuencia.

Acondicionador de señal

Incluye los circuitos de soporte para el transductor. Estos circuitos pueden proporcionarle la energía de excitación, circuito de equilibrio y elementos de calibración. Enseguida se presentan algunos aspectos en cuanto al acondicionamiento de señal en sistemas de adquisición y que deben ser tomados en cuenta, [18].

- **Filtrado**. El ambiente en que se llevan a cabo las mediciones eléctricas puede ocasionar que señales indeseables de interferencia se acoplen en el sistema de medición. Para evitar que esta interferencia ocasione errores en los datos de la medición se emplean filtros, los cuales son diseñados para rechazar rangos de frecuencia específicos. Se puede implementar circuitos de filtro con solo resistencias, capacitancias e inductancias (filtros pasivos), o bien, empleando transistores y amplificadores operacionales (filtros activos). Muchos transductores tienen una velocidad de transferencia de información baja y sus anchos de banda máximos son de cerca de 10 Hz por lo que el filtrado de ruido de las señales de estos transductores se realiza con el uso de filtros paso baja ya que responden perfectamente a señales de muy baja frecuencia y tienen gran atenuación a altas frecuencias.

- **Desviación y conversión de nivel**. Puede ser que la salida del transductor esté en una forma (cambio de voltaje o resistencia), mientras que el dispositivo de lectura podría necesitar la señal en otra forma (una corriente de 4 a 20 mA, por ejemplo). También podría ser que la señal tuviera una variación pequeña en comparación con un valor grande. En estos casos es necesario algún método de acondicionamiento de señal que la altere correctamente para hacerla compatible para la interacción con los siguientes elementos del sistema. Algunas de las técnicas de desviación y conversión de nivel en los sistemas de adquisición son:

  - El empleo de circuitos sujetadores para desplazar las pequeñas variaciones de voltaje con respecto al valor grande fijo a partir del cual varían (por ejemplo 0-5 V, 1-5 V).
  - Un convertidor modular de voltaje a corriente de 4 a 20 mA para cambiar las señales de la forma de voltaje a corriente.

- **Amortiguamiento**. La impedancia de salida de los transductores puede ser fuente de problemas en ciertas aplicaciones de sistemas. Por ejemplo, los transductores de alta impedancia de salida pueden originar errores no deseados de carga o pueden afectar los tiempos de asentamiento de los multiplexores del sistema. Las impedancias desequilibradas pueden conducir a errores de interferencia de modo común. se usan circuitos tanto pasivos como activos como elementos acondicionadores de señal de transformación de impedancia (o amortiguadores) para superar esos problemas.

Multiplexor

Acepta múltiples entradas analógicas y las conecta secuencialmente a un instrumento de medición. En la conversión analógica a digital es común multiplexar varias entradas analógicas en un solo canal ántes de pasar al convertidor A/D. En la Figura 1.2 se muestra de manera general un sistema de conversión A/D con entradas multiplexadas, donde los interruptores conectan las entradas analógicas a un bus común, el cual va a un solo convertidor A/D que sirve para todos los canales. Las entradas analógicas se conmutan de manera secuencial o programada al bus por medio de un circuito de control y selección de canales, [17].

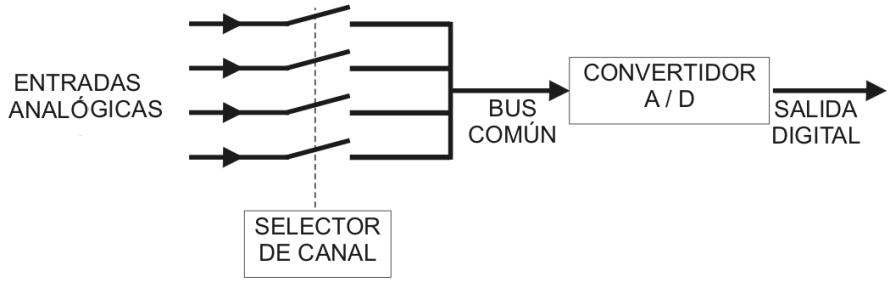

Figura 1.2: Sistema de conversión A/D multiplexado.

Convertidor analógico a digital (A/D)

Una magnitud analógica es aquella que tiene valores continuos dentro de un determinado rango, a diferencia del conjunto de valores discretos que tienen las señales digitales. En la naturaleza muchas cantidades son analógicas, por ejemplo, las cantidades físicas tales como temperatura, presión, tiempo y velocidad, (Figura 1.3).

Un convertidor A/D toma un voltaje de entrada analógico y después de cierto tiempo produce un código de salida digital que representa la entrada analógica. La salida del convertidor A/D se puede desplegar visuálmente y estar disponible como voltaje en pasos discretos para su procesamiento posterior o grabación en un registro digital, [18].

Registro digital

Registra la información digital en algún medio de almacenamiento como pueden ser cintas magnéticas o elementos de memoria de estado sólido.

1.4.2. Conversión A/D

Para realizar la tarea de conversión, el convertidor A/D tiene que efectuar procesos como muestreo,cuantización y codificación, [18], [19], [20], [21].

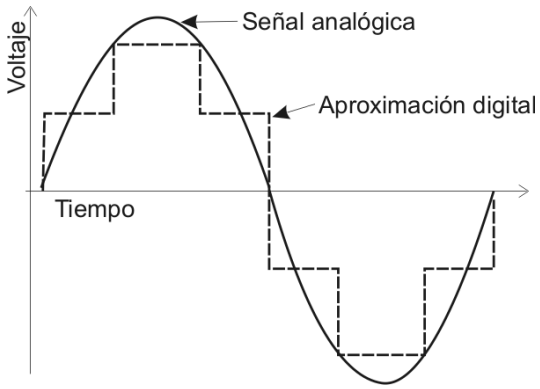

Figura 1.3: Señales analógica y digital.

En la Figura 1.4 se muestra el proceso de conversión de señales analógicas a digitales, [19].

Muestreo de una señal analógica

Para convertir una señal analógica en digital, el primer paso consiste en realizar un muestreo (sampling) de ésta, lo que equivale a tomar diferentes muestras de tensiones o voltajes en diferentes puntos de la señal analógica. La frecuencia a la que se realiza el muestreo se denomina razón, tasa o también frecuencia de muestreo y se mide en kilohertz, (Figura 1.4a).

Durante el proceso de muestreo se asignan valores numéricos equivalentes a la tensión o voltaje existente en diferentes puntos de la señal, con la finalidad de realizar a continuación el proceso de cuantización.

Cuantización de una señal muestreada

Una vez realizado el muestreo, el siguiente paso es la cuantización de la señal analógica. Para esta parte del proceso los valores continuos de la señal se convierten en series de valores numéricos decimales discretos correspondientes a los diferentes niveles o variaciones de voltajes que contiene la señal analógica original, (Figura 1.4b).

Como ejemplo se tiene una señal analógica cuyos valores van de 0 a +5 V. Si se desea convertir esta señal a forma digital y la salida requerida es una señal de 4 bits, entonces se puede representar $2^n = 16$ valores diferentes, 0 a 15. La resolución de la conversión (Ecuación 1.1), será $5V/15 = 1/3V$. Por lo tanto, una señal analógica de 0V

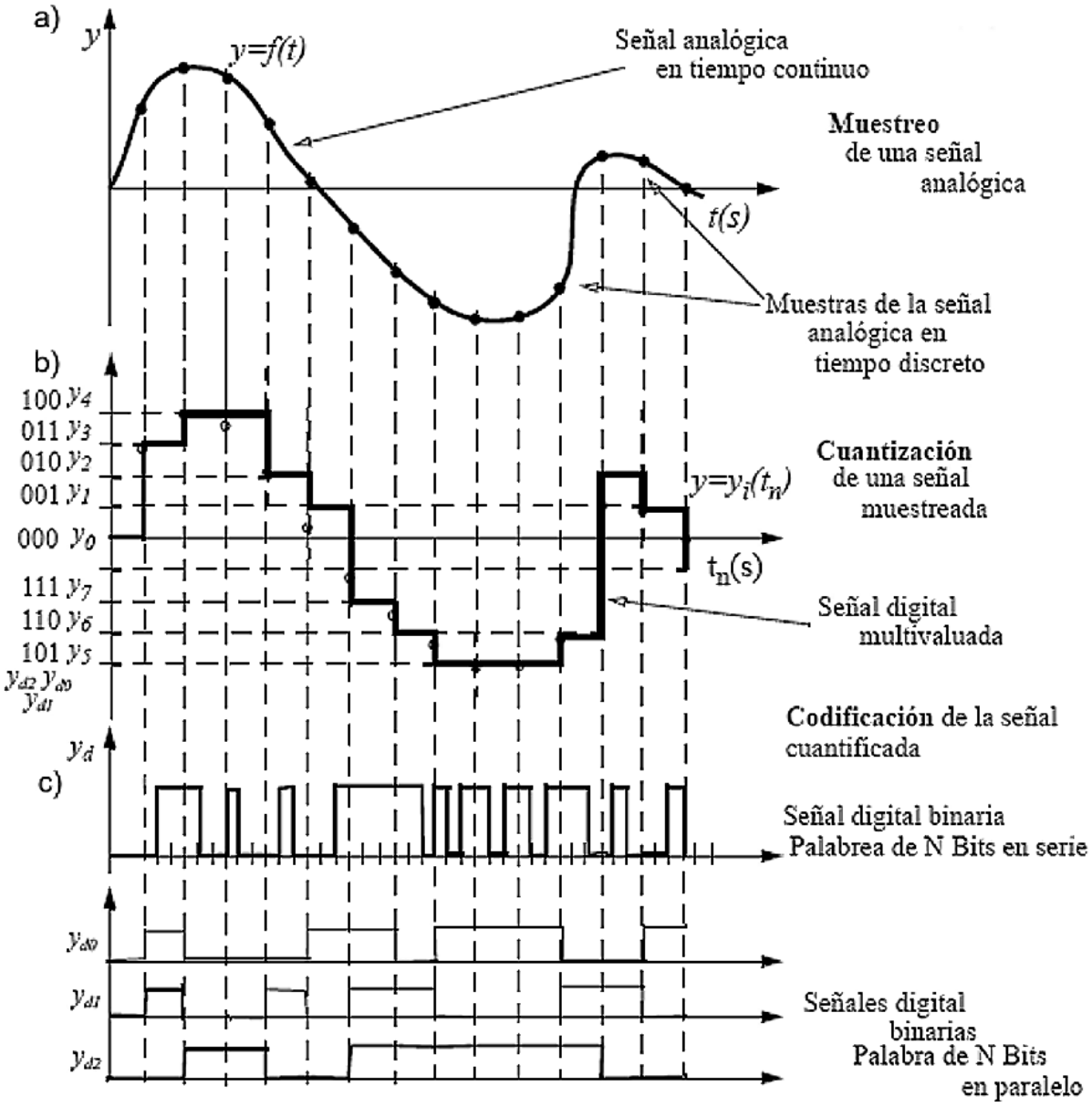

Figura 1.4: Proceso conversión A/D. a)muestreo, b)cuantización, c)codificación.

Sistema Decimal	Código Binario
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Tabla 1.1: Codificación con 3 bits.

estará representada por 0000, 1/3V por 0001, 3V por 1001 y 5V por 1111.

$$resolucion = \frac{V_{ref}}{2^n - 1} \tag{1.1}$$

Todos los números de muestra anteriores son múltiplos del incremento básico (1/3V). Pero para los valores que están entre estos niveles incrementales sucesivos, por ejemplo, para el caso de un nivel analógico de 3.1V, esta entre 9/3 y 10/3, pero como esta más cercano a 9/3 lo tratamos como si fuera 3V y lo codificamos como 1001. Este proceso se denomina cuantización. Obviamente, hay errores que reciben el nombre de errores de cuantización. Si se usan mas bits para representar (o simplemente codificar) una señal analógica se reducen los errores de cuantización al aumentar la resolución, pero se requieren circuitos más complejos.

Codificación de una señal cuantificada

Después de realizada la cuantización, los valores de las tomas de voltajes se representan numéricamente por medio de códigos y estándares previamente establecidos. Lo más común es codificar la señal digital en código numérico binario, (Figura 1.4c).

Como ejemplo en la Tabla 1.1 se muestra los valores numéricos del 0 al 7, pertenecientes al sistema decimal y sus equivalentes en código numérico binario. En esta tabla se puede observar que utilizando sólo tres bits por cada número en código binario, se pueden representar ocho niveles o estados de cuantización.

Convertidor A/D de aproximaciones sucesivas

Aunque existen diversos tipos se describe de manera breve el funcionamiento del convertidor de aproximaciones sucesivas ya que el microcontrolador utilizado cuenta con un convertidor A/D de este tipo.

Los convertidores de aproximaciones sucesivas usan un registro con lógica de control que modifica el contenido del registro bit a bit hasta que los datos del registro son el equivalente digital de la entrada analógica.

El tiempo de conversión de los convertidores de aproximaciones sucesivas de n bits requieren n ciclos de reloj para realizar su conversión sin importar la magnitud del voltaje que esta presente en su entrada, esto se debe a que los circuitos de control tienen que ensayar un 1 lógico en cada posición del bit para ver si se necesita o no, es por esto que los convertidores de aproximaciones sucesivas tienen tiempos de conversión muy rápidos, [20], [21], [22].

Su funcionamiento es similar al algoritmo que usamos las personas para acertar un número dentro de un rango: Primero intentamos con el valor 1/2 del máximo. Si nos contestan que es menor, lo intentamos con el valor 1/4 del máximo; y así sucesivamente vamos acotando el número hasta acertar.

Por ejemplo, la primera comparación se deberá efectuar entre la tensión de entrada V_{in} y la tensión $V_h = V_{ref}/2$, correspondiente a la palabra 1000, si V_h es mayor o igual a V_{in}, se determina que el bit de mayor peso debe ser uno, pero si $V_h < V_{in}$, se debe memorizar un cero en el registro de salida, en el siguiente impulso de reloj se efectúa una segunda comparación de V_h correspondiente a la palabra 1100, si la comparación anterior había dado positiva, o se compara contra 0100 en caso contrario, la salida del comparador determina el valor que debe memorizarse con un bit de peso $V_{ref}/4$, configurándose de esta forma, la palabra de salida digital una vez efectuadas las n comparaciones sucesivas. El registro de aproximaciones sucesivas esta constituido básicamente por un contador y un decodificador, que efectúa el direccionamiento de los n biestables del registro.

Capítulo 2

Herramientas básicas de desarrollo

2.1. Software

Para el desarrollo del proyecto se utilizan varios paquetes de software con el fin de cumplir el objetivo planteado. Para proponer los rangos adecuados de las variables, se emplea MATLAB para obtener gráficas de las variables que se están considerando y Jflap para el diseño de automata y su representación en un diagrama de estados.

Una vez determinados los rangos y las transiciones de los estados es posible implementar el algoritmo de control en el PIC16F877 mediante el uso de MPLAB IDE, una herramienta que permite desarrollar aplicaciones para microcontroladores.

La supervisión del proceso se realiza gracias a LabVIEW, programa que nos permite realizar la comunicación serial con una PC de una manera sencilla y mediante un ambiente gráfico.

2.1.1. MATLAB

MATLAB es la abreviatura de Matrix Laboratory (laboratorio de matrices). Es un programa de matemáticas creado por The MathWorks en 1984. Está disponible para las plataformas Unix, Windows y MAC. Usa un lenguaje de programación creado en 1970 para proporcionar un sencillo acceso al software de matrices LINPACK y EISPACK. También tiene su propio compilador, [23].

MATLAB dispone también en la actualidad de una amplia gama de programas de apoyo especializados, denominados *Toolboxes*, que extienden significativamente el número de funciones incorporadas en el programa principal. Estos Toolboxes ayudan a cubrir necesidades de casi todas las áreas principales en el mundo de la ingeniería y la simulación, destacando entre ellos el toolbox de proceso de imágenes, señal, control robusto, estadística, análisis financiero, matemáticas simbólicas, redes neuronales, lógica

difusa, identificación de sistemas, simulación de sistemas dinámicos, etc. Es un entorno de cálculo técnico, que se ha convertido en estándar de la industria con capacidades no superadas en computación y visualización numérica. De forma coherente, integra los requisitos claves de un sistema de computación técnico: cálculo numérico, gráficos, herramientas para aplicaciones especificas y capacidad de ejecución en múltiples plataformas, [24], [25].

MATLAB es un software muy usado en universidades, centros de investigación y por ingenieros, ya que puede utilizarse para casi cualquier caso o problema que requiera de cálculos, entre ellos se puede utilizar para desarrollo de algoritmos, modelado, simulación y desarrollo de prototipos, análisis de datos, exploración y visualización, para aplicaciones las cuales incluyen el desarrollo de interfaces gráficas para el usuario, [23].

Tiene un rango completo de funciones para preprocesar datos para análisis, incluyendo:

- Decimando
- Secciones de datos
- Promedio
- Proceso de umbrales
- Filtrado de numerosas operaciones para manipular arreglos multidimensionales, incluyendo reticulación e interpolación de datos, están también disponibles.

Todos los atributos de los gráficos de MATLAB son personalizables, desde los rótulos de ejes al ángulo de la fuente de luz en las superficies 3-D. Los gráficos están integrados con las capacidades de análisis, de modo que es posible mostrar gráficamente cualquier conjunto de datos sin editar, ecuación o resultado funcional.

Las ventanas de despliegue gráfico son muy similares, en las cuales el énfasis de la presentación se pone en la gráfica generada y no en el entorno de trabajo, pero debemos recordar que como todo este tipo de aplicaciones su desarrollo está orientado al logro de un objetivo especifico como es el resolver modelos matemáticos, [25], [26].

2.1.2. MPLAB IDE

MPLAB IDE es un software que junto con un emulador y un programador, forman un conjunto de herramientas de desarrollo muy completo para el trabajo y diseño con los Microcontroladores PIC.

Incorpora todas las utilidades necesarias para la realización de cualquier proyecto, y si no se dispone de un emulador, el programa permite editar el archivo fuente en lenguaje ensamblador, además de ensamblarlo y simularlo en pantalla, pudiendo ejecutarlo posteriormente y verlo paso a paso y ver como evolucionaría de forma real tanto sus registros internos, la memoria RAM y EEPROM de usuario como la memoria de programa, según se fueran ejecutando las instrucciones. Además el entorno es el mismo que si se estuviera utilizando un emulador, [27].

Este software es muy fácil de utilizar debido a que trabaja con un ambiente de ventanas, y además es de distribución gratuita a través de la página de Microchip, [27]. Un inconveniente es que no es posible utilizar lenguaje de programación C al instalar el programa. Para poder utilizar lenguaje C es necesario contar también con PICCLITE de HI-TECH Software [28], el cual es software de desarrollo para microcontroladores mediante lenguaje C pero en un ambiente MSDOS, por lo que deben integrarse las dos herramientas para utilizar el compilador de PICCLITE desde MPLAB IDE, conseguiéndose así los beneficios de estas dos herramientas. Una versión de prueba de PICCLITE se puede conseguir visitando la página de HI-TECH Software, [28], y aunque solo permite utilizar 20 % de la memoria de programa del PIC16F877, es suficiente para la implementación del algoritmo de control de las cortinas de ventilación.

2.1.3. Jflap

Es un sistema gráfico implementado en Java que sirve de apoyo al aprendizaje de muchos de los conceptos básicos relacionados con la Teoría de Lenguajes Formales y Autómatas. Las características generales de JFLAP es que incluye expresiones regulares para un AFN, y un AFD.

JFLAP dispone de una interfaz gráfica a través de la cual el usuario puede introducir autómatas así como visualizar gráficamente la salida de los diferentes algoritmos. La versión actual contiene diferentes mejoras que se han ido incorporando a las sucesivas versiones de este, de tal manera de que JFLAP incluye nuevas características permitiendo así la representación de diferentes tipos de autómatas, [29].

2.1.4. LabVIEW

LabVIEW es una herramienta gráfica de prueba, control y diseño mediante la programación.

El lenguaje que usa se llama lenguaje G. Este programa fue creado por National Instruments (1976) para funcionar sobre máquinas MAC. Salió al mercado por primera vez en 1986. Ahora está disponible para las plataformas Windows, UNIX, MAC y Li-

nux y actualmente está disponible la versión 7.1 (desde julio de 2004). Los programas hechos con LabVIEW se llaman VI (Virtual Instrument), lo que da una idea de uno de sus principales usos: el control de instrumentos.

LabVIEW permite combinar el software con todo tipo de hardware, tanto propio (tarjetas de adquisición de datos, PAC, Visión, y otro Hardware), como de otras empresas, [23].

Principales usos

Es usado principalmente por ingenieros y científicos para tareas como:

- Adquisición de datos
- Control de instrumentos
- Automatización industrial o PAC (Controlador de Automatización Programable)
- Diseño de control: Diseño de prototipos rápido y hardware-en-el-bucle (HIL)

Principales características

Su principal característica es la facilidad de uso, personas con pocos conocimientos en programación pueden hacer programas relativamente complejos, imposibles para ellos de hacer con lenguajes tradicionales. También es muy rápido hacer programas con LabVIEW y cualquier programador, por experimentado que sea, puede beneficiarse de él. Con LabVIEW pueden crearse programas de miles de VIs (páginas de código) para aplicaciones complejas, programas de automatizaciones de decenas de miles de puntos de entradas/salidas, etc. Incluso existen algunas prácticas de programación para optimizar el rendimiento y la calidad de la programación.

Presenta facilidades para el manejo de:

1. Interfaces de comunicaciones:
   - Puerto serie
   - Puerto paralelo
   - TCP/IP,UDP, DataSocket
   - IrDA
   - Bluetooth

- USB

2. Capacidad de interactuar con otras aplicaciones:
   - dll
   - ActiveX
   - MATLAB
   - Simulink
3. Herramientas para el procesado digital de señales.
4. Visualización y manejo de gráficas con datos dinámicos.
5. Adquisición y tratamiento de imágenes.
6. Control de movimiento.
7. Tiempo Real estrictamente hablando.
8. Sincronización.

El Programa en LabVIEW

Como se dijo es una herramienta gráfica de programación, esto significa que los programas no se escriben, sino que se dibujan.

Un programa se divide en panel frontal y diagrama de bloques. El panel frontal es la interfaz con el usuario, en él se definen los controles e indicadores que se muestran en pantalla. El diagrama de bloques es el programa propiamente dicho, donde se define su funcionalidad, aquí se colocan iconos que realizan una determinada función y se interconectan entre si, [23], [30].

2.2. Hardware

Para el diseño del autómata es necesario el uso de varios elementos y periféricos, tanto para la medición y la supervisión de las variables de las cuales depende el autómata, como para el control de los actuadores que pondrán en funcionamiento el sistema mecánico de cortinas.

Estos elementos son:

1. Sistema mínimo.
2. Microcontrolador PIC16F877.
3. Pantalla de cristal líquido (LCD).
4. Interfaz de comunicación serial RS-232.
5. Sensores.

En la Figura 2.1 se muestra el diagrama a bloques del sistema para controlar la apertura y cierre de cortinas del invernadero. Como se puede observar, en cada puerto de entrada y de salida (E/S) del sistema mínimo se indica el nombre de las terminales a la que se conecta cada periférico.

2.2.1. Sistema mínimo

Un sistema mínimo es un dispositivo que ayuda a una simplificación de elementos que interactúan en un proceso. Para la conexión de estos elementos se utiliza el diseño de un circuito que funciona con un microcontrolador PIC16F877 de Microchip.

Las principales características del sistema mínimo son:

- Entrada de alimentación del circuito de 5 V regulados.
- Circuito oscilador de cristal de 20 MHz.
- Zócalo de 40 pines para montar el microcontrolador PIC16F877.
- Circuito para reset (RST).
- Entradas analógicas.
- Entradas y salidas digitales (EXPANSIÓN, PUERTO D).
- Puerto de comunicación serial (SCI) y paralelo (PUERTO D).
- Conector para LCD (DISPLAY) y control de contraste.

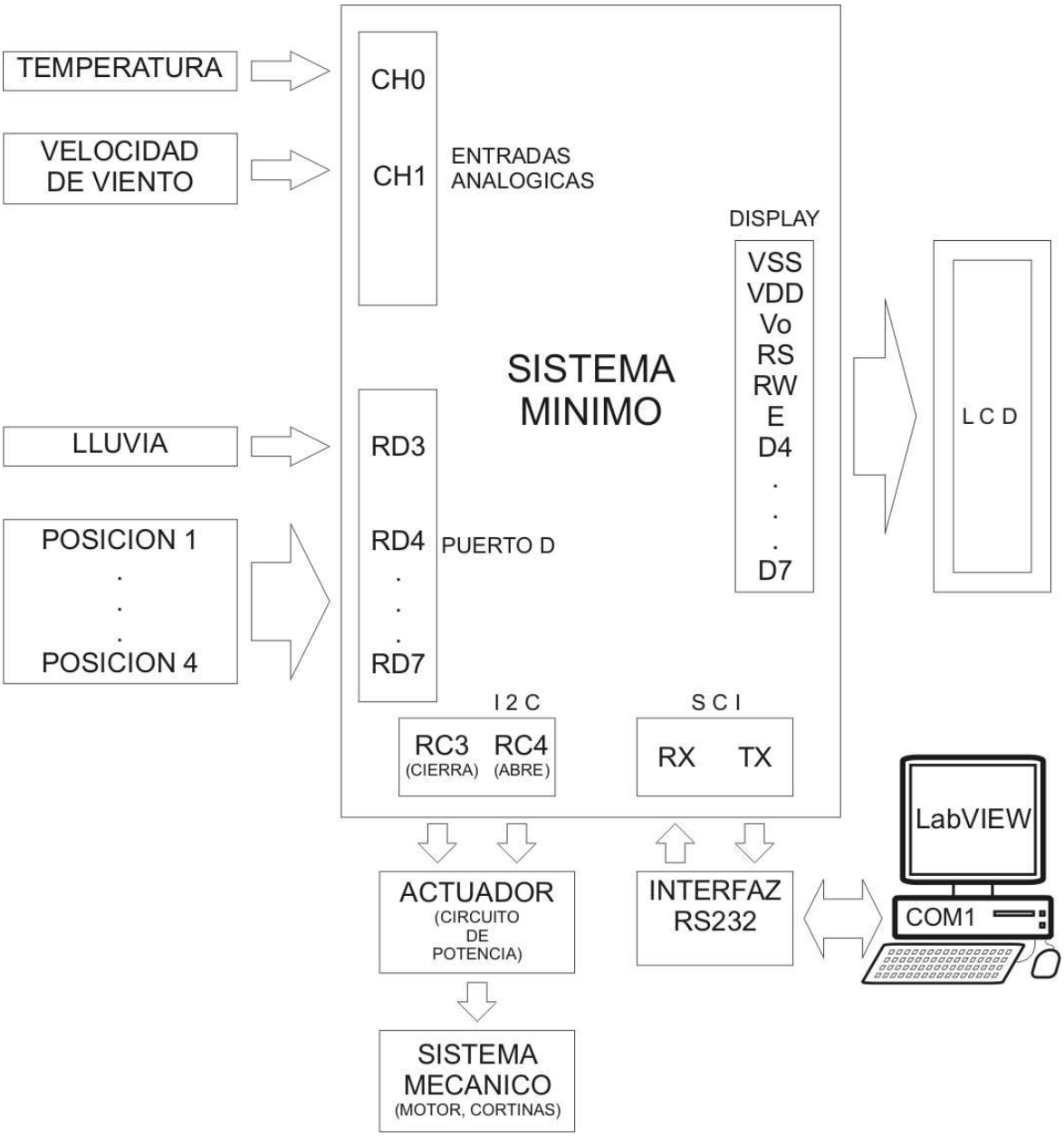

Figura 2.1: Diagrama general del sistema de control.

En la Figura 2.2 se muestra el nombre y la distribución de los puertos de E/S del sistema mínimo.

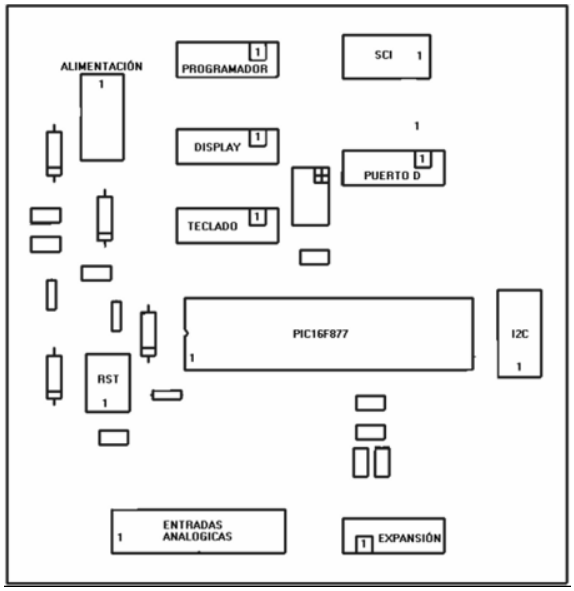

Figura 2.2: Diagrama del sistema mínimo basado en el PIC16F877.

2.2.2. Microcontrolador PIC16F877

Un microcontrolador es un circuito integrado programable que contiene todos los componentes de un computador. Se emplea para controlar el funcionamiento de una tarea determinada y debido a su reducido tamaño suele ir incorporado en el propio dispositivo al que gobierna.

El alto rendimiento y elevada velocidad de los microcontroladores PIC, se debe principalmente a tres de sus principales características, [31], [32].

1. Arquitectura de tipo Harvard. En ésta, la unidad de proceso central se conecta de forma independiente y con buses distintos con la memoria de instrucciones y con la de datos. La arquitectura Harvard permite a la CPU acceder simultáneamente a las dos memorias.

2. Técnica de segmentación para la ejecución de las instrucciones. Esta técnica permite al procesador realizar al mismo tiempo la ejecución de una instrucción y la búsqueda del código de la siguiente. De esta forma se puede ejecutar cada instrucción en un ciclo (un ciclo de instrucción equivale a cuatro ciclos de reloj).

3. Procesador RISC (Computador de Juego de Instrucciones Reducido). Los modelos de la gama baja disponen de un repertorio de 33 instrucciones de 12 bits de longitud, 35 instrucciones de 14 bits los de la gama media y casi 60 instrucciones de 16 bits los de gama alta.

La Figura 2.3 muestra el diagrama interno del microcontrolador PIC16F877 donde se pueden ver los diferentes bloques, los puertos de entrada/salida y los buses que los conectan, [33].

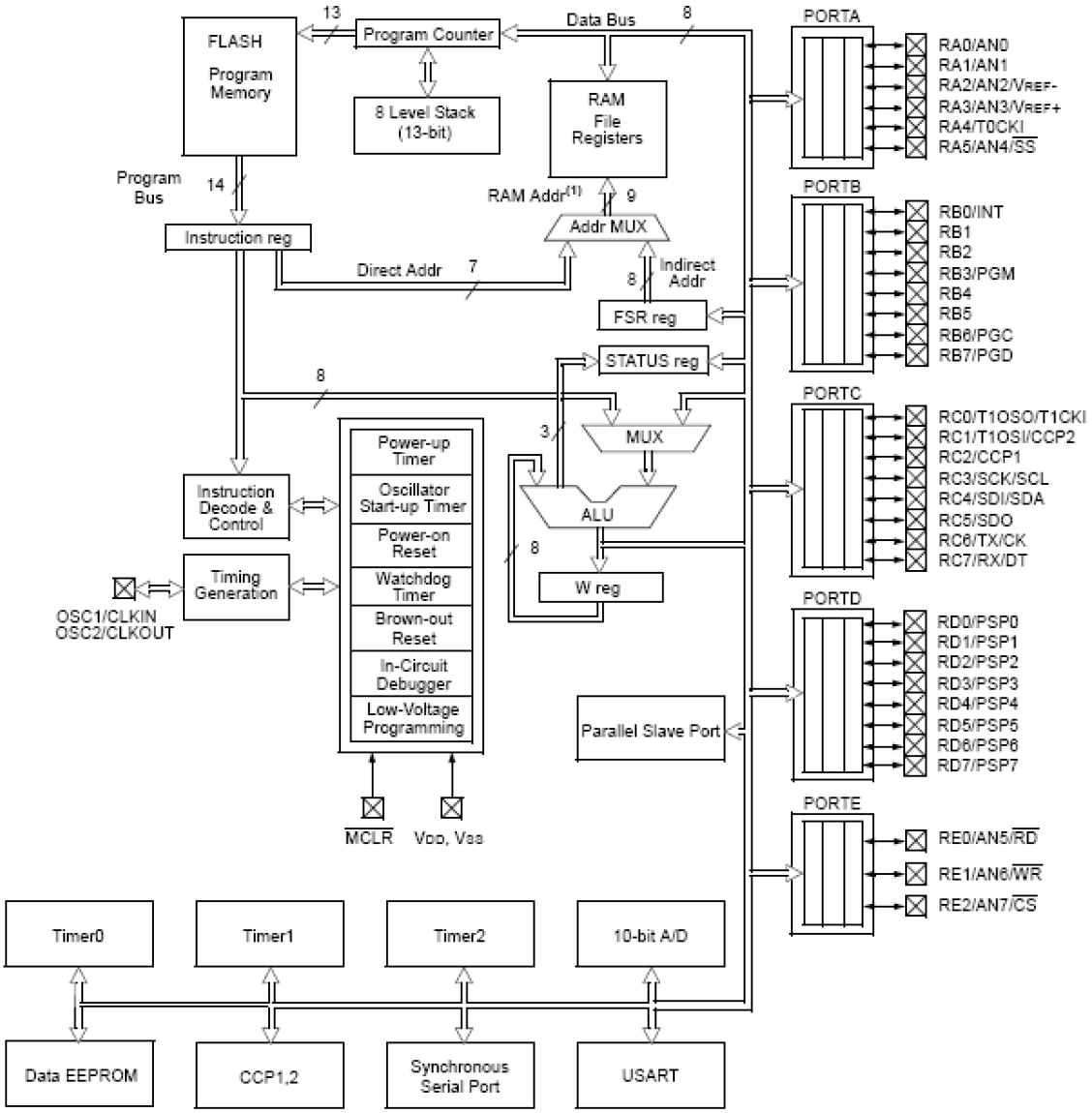

Figura 2.3: Diagrama del PIC16F877.

Se elige este microcontrolador debido a que cuenta con 8 canales de entrada analógicos y un bloque de conversión A/D con una resolución de 10 bits, suficiente para el tipo de señales que se van a manejar. El espacio de memoria para programa es suficiente para la mayoría de rutinas y cuenta además con una memoria para datos que puede ser modificada de manera rápida y eficiente. Tiene implementado un módulo de comunicación serial que puede ser configurado con distintos modos de transmisión.

Descripción de terminales

Cada uno de las terminales o pines de los puertos A, B, C, D y E pueden funcionar como entradas o salidas digitales, aunque la mayoría pueden desempeñar otras funciones.

En la Figura 2.4 se muestra la distribución de terminales y en la Tabla 2.1 se enumera cada una de sus funciones, [33].

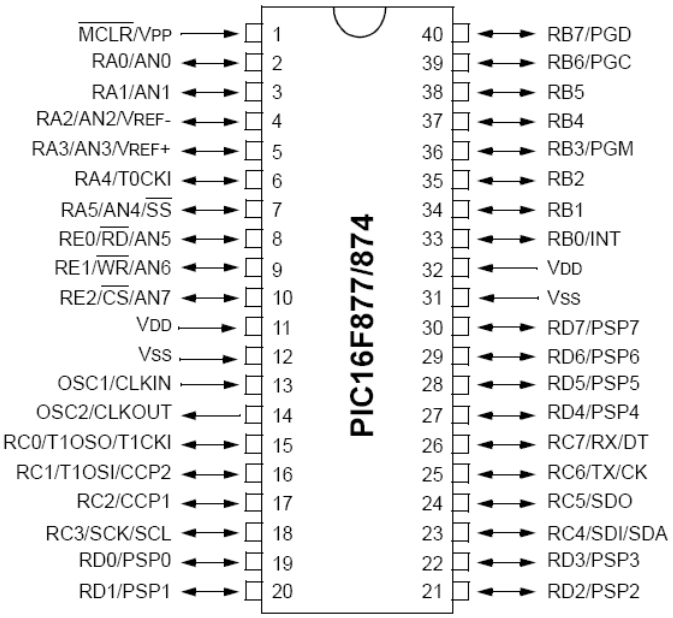

Figura 2.4: Distribución de terminales del PIC16F877.

Pin	Nombre	Descripción
1	MCLR/VPP	Entrada de reset maestro o de voltaje de programación
2	RA0/AN0	Entrada analógica 0
3	RA1/AN1	Entrada analógica 1
4	RA2/AN2/VREF-	Entrada analógica 2 o voltaje negativo de referencia analógico
5	RA3/AN3/VREF+	Entrada analógica 3 o voltaje positivo de referencia analógico.
6	RA4/T0CKI	Entrada de reloj para el Timer0
7	RA5/SS/AN4	Entrada analógica o selector de esclavo para el puerto serial síncrono
8	RE0/RD/AN5	Control de lectura para el puerto paralelo esclavo o entrada analógica 5
9	RE1/WR/AN6	Control de escritura para el puerto paralelo esclavo o entrada analógica 6
10	RE2/CS/AN7	Control de selección para el puerto paralelo esclavo o entrada analógica 7
11,32	VDD	Voltaje de alimentación positivo
12,31	VSS	Referencia a tierra (Ground)
13	OSC1/CLKIN	Entrada de oscilador de cristal/reloj externo
14	OSC2/CLKOUT	Salida de oscilador de cristal.
15	RC0/T1OSO/T1CKI	Salida del oscilador del Timer1 o entrada de reloj del Timer1
16	RC1/T1OSI/CCP2	Entrada del oscilador del Timer1 o entrada de captura 2/salida de comparador 2/salida RISC2
17	RC2/CCP1	Entrada de captura 1/ salida de comparador 1/ salida PWM1
18	RC3/SCK/SCL	Entrada de reloj para la comunicación serial síncrona o salida para los modos SPI (Interfaz de puerto serial) e I2C
19-23	RD0-3/PSP0-3	Puerto paralelo esclavo
23	RC4/SDI/SDA	Entrada de datos en modo SPI o E/S de datos en modo I2C
24	RC5/SDO	Salida de datos en modo SPI
25	RC6/TX/CK	Transmisión asíncrona del módulo USART o reloj en modo síncrono
26	RC7/RX/DT	Recepción asíncrona del módulo USART o datos en modo síncrono
27-30	RD4-7/PSP4-7	Puerto paralelo esclavo
33	RB0/INT	Entrada para interrupción externa
34,35	RB1,RB2	E/S digital
36	RB3/PGM	Entrada para programación con bajo voltaje
37,38	RB4,RB5	Pin para interrupción por algún cambio de valor en el puerto
39	RB6/PGC	Pin para interrupción por algún cambio de valor en el puerto o reloj durante programación serial
40	RB7/PGD	Pin para interrupción por algún cambio de valor en el puerto o datos durante programación serial

Tabla 2.1: Descripción de terminales del PIC16F877.

Memoria

La memoria se organiza en tres bloques principales, [33].

- Memoria de programa tipo FLASH, no volátil en la que se graba el programa de aplicación, y tiene una capacidad de de 8K palabras de 14 bits (Figura 2.5.)
- Memoria de datos RAM de 368 bytes. Se divide en 4 bancos y contienen los registros de propósito general y de función especial (Figura 2.6).
- Memoria de datos EEPROM, no volátil de 256 bytes. En esta memoria se pueden almacenar datos generados durante el funcionamiento del sistema para un manejo posterior.

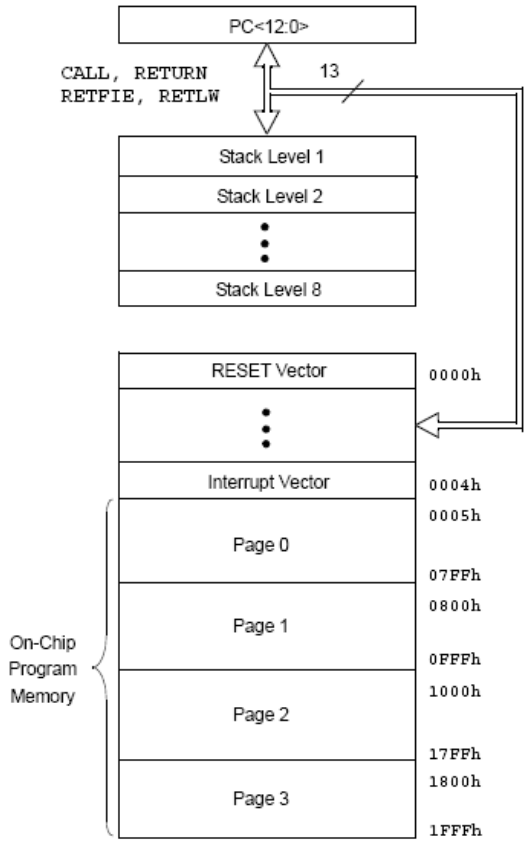

Figura 2.5: Memoria de programa, stack, vectores de reset e interrupciones.

Stack

El stack o pila (Figura 2.5), es una memoria interna dedicada, de tamaño limitado, separada de las memorias de datos y de programa, inaccesible al programador, y organizada en forma de pila, que es utilizada solamente, y en forma automática, para guardar las direcciones de retorno de subrutinas e interrupciones (anidado de funciones). Cada posición permite guardar una copia completa del PC (contador de programa). Como en toda memoria tipo pila, los datos son accedidos de manera tal que el primero que entra es el último que sale [34].

El PIC16F877 cuenta con una pila de 8 posiciones, por lo que al hacer un mayor anidamiento de funciones, ocurre un desbordamiento de la pila y se pierden las primeras direcciones del contador de programa almacenadas y no será posible retornar a ellas desde alguna subrutina.

Palabra de configuración

La palabra de configuración es una posición reservada de la memoria de programa FLASH, que ocupa la dirección 2007h y que solo es accesible durante la programación del PIC, [33]. Su modificación nos permite cambiar parámetros del funcionamiento del microcontrolador tales como:

- Código de protección de memoria de programa (CP1:CP0).
- Modo Depurador en circuito (DEBUG).
- Permiso de escritura en la memoria FLASH (WRT).
- Código de protección de la memoria EEPROM (CPD).
- Permiso para programación en bajo voltaje (LVP).
- Permiso para reset por baja tensión (BODEN).
- Permiso para el Timer de conexión de alimentación (PWRTE).
- Permiso de Timer del perro guardián (WDTE).
- Tipo de oscilador (FOSC1:FOSC0).

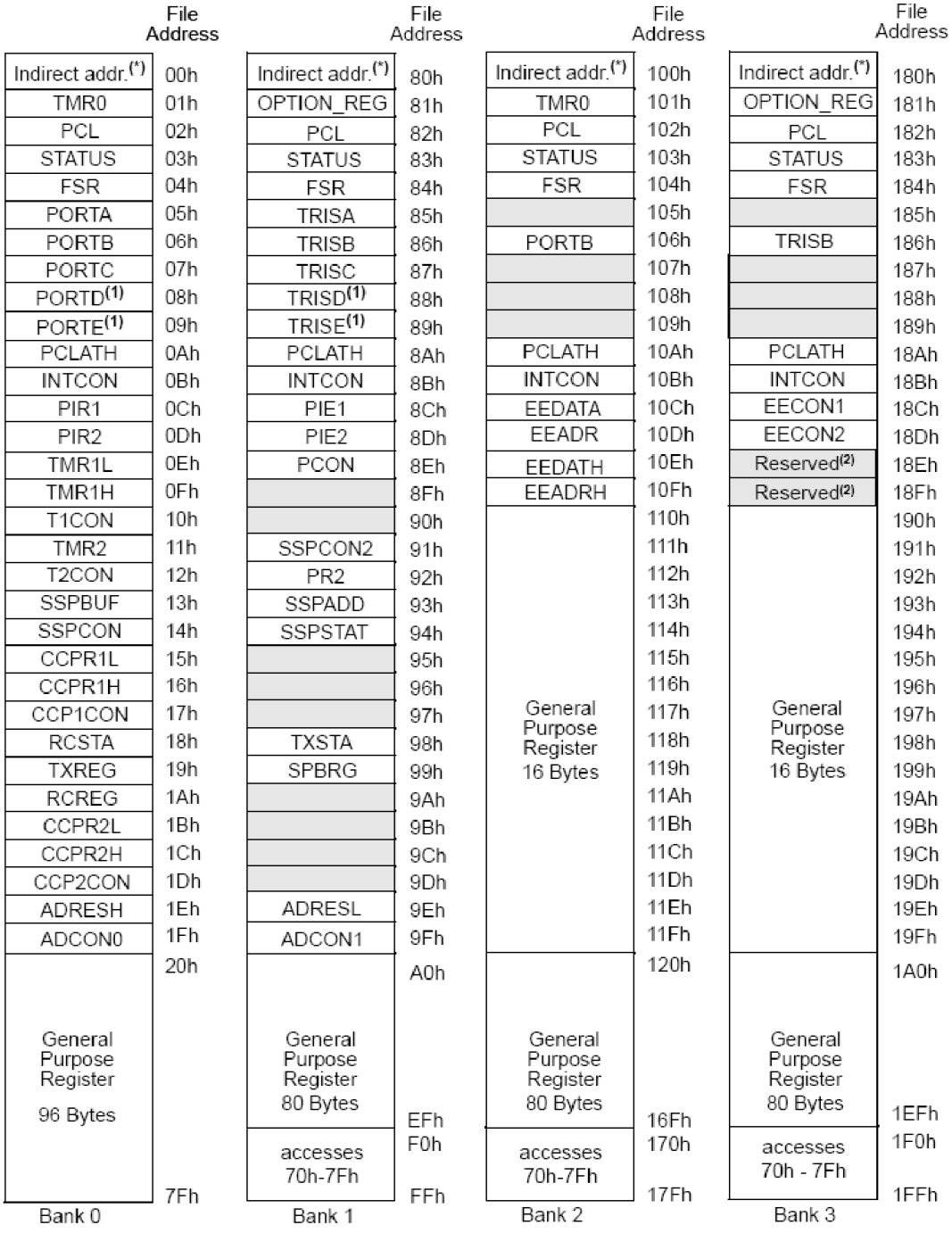

Figura 2.6: Mapa de registros en la memoria RAM.

Interrupciones internas/externas

El PIC cuenta con catorce fuentes distintas de interrupciones que pueden ser activadas por eventos externos e internos, [33]. Entre ellas se tienen:

1. Interrupción por desborde del Timer0, Timer1, Timer2.
2. Interrupción externa mediante el pin 0 del puerto B (RB0/INT).
3. Interrupción por un cambio en los 4 bits más significativos del puerto B.
4. Interrupción de lectura/escritura en el puerto paralelo.
5. Interrupción por fin de conversión analógico/digital.
6. Interrupción por fin de recepción en el módulo de comunicación serial.
7. Interrupción por fin de transmisión desde el módulo de comunicación serial.
8. Interrupción por cada módulo de captura/comparación/PWM (CCP1, CCP2).
9. Interrupción por operación de escritura en la memoria EEPROM.
10. Interrupción por transferencia en el puerto de comunicación serial síncrona.
11. Interrupción por colisión en el bus de comunicación serial síncrona.

Registros de función especial

Mediante estos registros se puede configurar o conocer el funcionamiento de cada uno de los recursos del microcontrolador. Se dividen en registros para uso de CPU y registros para uso de periféricos. Están ubicados en las primeras posiciones de cada uno de los bancos de memoria y algunos se encuentran en dos o más bancos debido a su importancia (Figura 2.6), [33].

Registro de control de interrupciones

Es un registro de lectura/escitura y contiene bits de habilitación o banderas de señalización de interrupciones para desborde del Timer 0, cambio de estado en los 4 bits más significativos del puerto B y por activación del pin RB0/INT.

INTCON

GIE	PEIE	TMR0IE	INTE	RBIE	TMROIF	INTF	RBIF

bit 7: **GIE**: Habilitación global de interrupciones.
bit 6: **PEIE**: Habilita interrupción por periféricos.
bit 5: **TMR0IE**: Habilita interrupción por desborde del Timer 0.
bit 4: **INTE**: Habilita interrupción externa en RB0/INT.
bit 3: **RBIE**: Habilita interrupción por cambio en el puerto B.
bit 2: **TMR0IF**: Bandera de interrupción por desborde del Timer 0.
bit 1: **INTF**: Bandera por interrupción externa en RB0/INT.
bit 0: **RBIF**: Bandera de interrupción por cambio en el puerto B.

Registro para habilitación de interrupciones 1

Contiene los bits de habilitación para las interrupciones por periféricos. El bit PEIE de INTCON debe estar activado (PEIE=1).

PIE1

PSPIE	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE

bit 7: **PSPIE**: Habilita interrupción por lectura/escritura en puerto paralelo.
bit 6: **ADIE**: Habilita interrupción para el convertidor A/D.
bit 5: **RCIE**: Habilita interrupción por recepción en puerto serial.
bit 4: **TXIE**: Habilita interrupción por transmisión en puerto serial.
bit 3: **SSPIE**: Habilita interrupción para puerto serial síncrono.
bit 2: **CCP1IE**: Habilita interrupción para módulo CCP1.
bit 1: **TMR2IE**: Habilita interrupción por Timer 2 .
bit 0: **TMR1IE**: Habilita interrupción por Timer 1.

Registro de banderas de interrupciones

En este registro, los bits actúan de señalizadores del momento en el que se origina la interrupción, independientemente de si está permitida o prohibida en correspondencia con el registro PIE1.

PIR1

PSPIF	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR1IF	TMR1IF

bit 7: **PSPIF**: Bandera de interrupción por lectura/escritura en puerto paralelo.
bit 6: **ADIF**: Bandera de interrupción por el conversión A/D.
bit 5: **RCIF**: Bandera de interrupción por recepción en puerto serial.
bit 4: **TXIF**: Bandera de interrupción por transmisión en puerto serial.
bit 3: **SSPIF**: Bandera de interrupción por puerto serial síncrono.
bit 2: **CCP1IF**: Bandera de interrupción por módulo CCP1.
bit 1: **TMR2IF**: Bandera de interrupción por Timer 2 .
bit 0: **TMR1IF**: bandera de interrupción por desborde del Timer 1.

Puertos bidireccionales de entrada/salida.

Se dispone de hasta cinco puertos que pueden ser configurados para funcionar como entradas o salidas. Los puertos A y E pueden funcionar también como entradas analógicas y el puerto D como esclavo paralelo. Los puertos B y C son de entrada/salida digital. Para más detalles consultar la referencia [33].

Algunos pines de los puertos están multiplexados con funciones alternas para algún periférico en el dispositivo, como el puerto de comunicación serial o salidas PWM. En general, cuando alguno de estos periféricos está habilitado esos pines no pueden ser utilizados como pines E/S de propósito general.

Puerto A.

El puerto A (PORTA), es un puerto de 6 bits y le corresponde un registro de dirección de datos (TRISA) como a cada uno de los demás puertos. Escribiendo un *1* en un bit de TRISA hace que el correspondiente bit en PORTA sea una entrada digital. Por el contrario, si se limpia un bit de TRISA, el correspondiente bit de PORTA será una salida digital.

Al leer el registro PORTA, se lee la información de cada uno de los bits y, al escribir en el registro se transfiere el valor deseado a cada pin del puerto. Cada operación de escritura implica que primero se deben leer el valor presente en los pines del puerto, se modifique el valor del registro y por último se manda ese valor a los pines.

El pin RA4 está multiplexado con la entrada de reloj del módulo Timer0. Otros pines están multiplexados con entradas analógicas o entradas de voltaje de referencia analógico. La operación de estos bits se selecciona poniendo a *0* ó *1* los bits de control del registro ADCON1 (registro de control A/D 1). En la Tabla 2.3 están las posibles configuraciones para los pines del puerto A/D de acuerdo a la combinación de los bits

menos significativos de ADCON1.

Hay un par de consideraciones importantes al utilizar este puerto:

1. Al existir un reset por encendido del PIC, estos pines son configurados como entradas analógicas y son leídos como *0*, por lo que se deben configurar por software al comienzo del programa.
2. Al utilizar los pines del puerto A como entradas analógicas (AN0-AN4), también se debe mantener en *1* los bits correspondientes del registro TRISA para que funcionen como salidas.

Puerto B.

El puerto B (PORTB) es un puerto bidireccional de 8 bits y su registro de dirección de datos correspondiente es el TRISB. Funciona de la misma manera que el TRISA, al poner un *1* ó un *0* a cada bit del registro se configura el pin correspondiente en el PORTB como entrada o salida respectivamente.

Tres pines del puerto B son multiplexados con la función de programación con bajo voltaje: RB3/PGM, RB6/PGC y RB7/PGD.

Cuatro de los pines de este puerto (RB4:RB7), cuentan con la posibilidad de activar una interrupción por algún cambio en el estado del pin. Solamente los pines configurados como entradas pueden activar esta interrupción y se hace al comparar el valor del pin con el valor de la lectura anterior del registro PORTB. Cuando algún valor no coincide durante la comparación se genera la interrupción con una bandera de interrupción: el bit RBIF en el registro INTCON en la posición 0.

Esta interrupción puede *reactivar* al dispositivo durante su operación en modo SLEEP y la bandera RBIF debe ser limpiada en la rutina de interrupción, primero leyendo o escribiendo al PORTB para que deje de verificar por algún cambio y enseguida limpiando la bandera.

El pin RB0 también cuenta con una interrupción externa. El pin correspondiente en TRISB debe ser puesto a *1* para funcionar como entrada y la interrupción se activa al recibir un pulso en el pin y puede ser configurada para activarse en el flanco de subida o en el flanco de bajada. La bandera de la interrupción se puede leer en el bit INTF del registro INTCON y debe ser limpiada durante la rutina de interrupción por software.

Puerto C.

Este es un puerto direccional de 8 bits y le corresponde el registro TRISC para direccionamiento de datos. El puerto C comparte funciones con algunos periféricos como

los módulo CCP1y CCP2, transmisión serial síncrona y asíncrona (SSP y USART).

Se debe tener especial cuidado en definir el valor de cada bit en TRISC para cada pin de PORTC cuando se habilitan algunos periféricos en este puerto. Algunos periféricos sobrescriben un bit en TRISC para hacer el pin correspondiente una salida o una entrada, como es el caso del módulo de comunicación USART, que hace que el pin RC6 funcione como salida (TX) y el bit RC7 como entrada (RX).

Puerto D.

El puerto D es bidireccional y tiene 8 bits. Cada pin puede ser configurado como entrada o salida de manera individual cambiando el bit correspondiente en el registro TRISD.

Un aspecto importante de este puerto es que puede ser configurado como un puerto entre el microprocesador de 8 bits (puerto paralelo esclavo). Para esto se le debe poner un *1* en el bit PSPMODE del registro TRISE (bit 4).

Puerto E.

El puerto E tiene solamente 3 pines que pueden ser configurados individualmente como entradas o salidas en los tres bits menos significativos de TRISE. Los pines de este puerto funcionan como entradas de control para el puerto del microprocesador cuando el modo PSP esta activado (bit PSPMODE). En este modo los pines del PORTE deben ser configurados como entradas en TRISE y en el registro ADCON1 se deben configurar como E/S digitales.

Estos pines también se utilizan como entradas analógicas al modificar los bits correspondientes en el registro ADCON1 y también deben ser configurados como entradas en TRISE (poniendo *1* al bit correspondiente a cada pin que funciona como entrada).

Convertidor A/D

El módulo de conversión Analógico/Digital dispone de 8 líneas en el microcontrolador PIC16F877. A través de las entradas analógicas se aplica la señal analógica a un condensador de muestreo y retención (sample and hold), la cual se introduce en el convertidor. El convertidor de aproximaciones sucesivas da como resultado una palabra de 10 bits. La resolución que tiene cada bit procedente de la conversión, tiene un valor que está en función de la tensión de referencia V_{ref}, de acuerdo con la Ecuación 1.1. Así, si la tensión de referencia positiva es 5V y la negativa es tierra (0V), entonces la resolución es de 4.88 mV/bit para un convertidor de 10 bits.

El funcionamiento del convertidor A/D requiere de la manipulación de cuatro registros:

1. ADRESH (parte alta del resultado de la conversión).
2. ADRESL (parte baja del resultado de la conversión).
3. ADCON0 (registro de control 0).
4. ADCON1 (registro de control 1).

En la pareja de registros ADRESH:ADRESL se deposita el resultado de la conversión, que al estar compuesta por 10 bits, solo son significativos 10 de los 16 bits de esa pareja de bytes.

El módulo A/D tiene la posibilidad de justificar este resultado, y la selección de este formato de justificación a la izquierda o derecha se realiza mediante la configuración del registro de control 1 (ADCON1). Los bits restantes (a los 10 de la conversión) se llenan con ceros. Cuando el convertidor A/D está en OFF y no se utilizan estos dos registros, pueden utilizarse como dos registros de 8 bits de propósito general [33].

Registro de control 0

El registro ADCON0 controla la operación del convertidor A/D y consta de 8 bits. Enseguida se definen cada uno de los bits y se muestra la estructura interna del registro.

ADCON0

ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	-	ADON

bit 7-6: **ADCS1:ADCS0**: Selección de reloj para la conversión A/D.
En la Tabla 2.2 se muestra los tipos de reloj disponibles, al igual que el dato que se ha de cargar en el bit ADCS2 del registro ADCON1.
bit 5-3: **CHS2:CHS0**: Seleccionan el canal analógico para la conversión A/D.
bit 2: **GO/DONE**: Indica que la conversión está en progreso. Cambia a *0* cuando la conversión ha terminado siempre que ADON este activado.
bit 1: No utilizado.
bit 0: **ADON**: Enciende el módulo de conversión A/D.

<ADCS2> ADCON1	<ADCS1:ADCS0> ADCON0	Reloj de Conversión
0	00	$Fosc/2$
0	01	$Fosc/8$
0	10	$Fosc/32$
0	11	F_{RC}
1	00	$Fosc/4$
1	01	$Fosc/16$
1	10	$Fosc/64$
1	11	F_{RC}

Tabla 2.2: Tabla de selección de bits para cambiar el reloj de conversión.

Registro de control 1

Este registro tiene la función de configurar las líneas del puerto A como entradas analógicas o entradas/salidas digitales y su estructura se muestra enseguida.

ADCON1

ADFM	-	-	-	PCFG3	PCFG2	PCFG1	PCFG0

bit 7: **ADFM**: Selección de formato de resultado de conversión A/D.
1=Justificado a la derecha. Los 6 bits más significativos de ADRESH son *0*
0=Justificado a la izquierda. Los 6 bits menos significativos de ADRESL son *0*.
bit 6-4: No utilizados.
bit 3-0: **PCFG3:PCFG0**: Bits de control de configuración del puerto A/D.

El bit más significativo ADFM de este registro selecciona el formato del resultado de la conversión. Si es *1*, los 2 bits más significativos del resultado de la conversión serán alojados en las dos primeras posiciones del registro ADRESH, dejando las 6 posiciones más significativas en *0*, mientras que los siguientes 8 bits se encontrarán en el registro ADRESL. Y si ADFM es igual a *0*, los 8 bits más significativos del resultado se encontrarán en el registro ADRESH y los 2 bits menos significativos se alojarán en las posiciones 6 y 7 del registro ADRESL, dejando las siguientes 6 igual a *0*.

Los restantes 4 bits (PCFG3:PCFG0) de ADCON1, se usan para configurar las líneas de los canales de entrada como entradas analógicas o entradas/salidas digitales (Tabla 2.3).

PCFG3: PCFG0	AN7 RE2	AN6 RE1	AN5 RE0	AN4 RA5	AN3 RA3	AN2 RA2	AN1 RA1	AN0 RA0	Vref+	Vref-
0000	A	A	A	A	A	A	A	A	VDD	VSS
0001	A	A	A	A	Vref+	A	A	A	RA3	VSS
0010	D	D	D	A	A	A	A	A	VDD	VSS
0011	D	D	D	A	Vref+	A	A	A	RA3	VSS
0100	D	D	D	D	A	D	A	A	VDD	VSS
0101	D	D	D	D	Vref+	D	A	A	RA3	VSS
011x	D	D	D	D	D	D	D	D	VDD	VSS
1000	A	A	A	A	Vref+	Vref-	A	A	RA3	RA2
1001	D	D	A	A	A	A	A	A	VDD	VSS
1010	D	D	A	A	Vref+	A	A	A	RA3	VSS
1011	D	D	A	A	Vref+	Vref-	A	A	RA3	RA2
1100	D	D	D	A	Vref+	Vref-	A	A	RA3	RA2
1101	D	D	D	D	Vref+	Vref-	A	A	RA3	RA2
1110	D	D	D	D	D	D	D	A	VDD	VSS
1111	D	D	D	D	Vref+	Vref-	D	A	RA3	RA2

Tabla 2.3: Bits 3-0 de ADCON1 para configuración del puerto A/D.

Comunicación serial

Este microcontrolador dispone de un módulo transmisor denominado USART (Universal Synchronous Asynchronous Receiver Transmitter) capaz de soportar la comunicación serial síncrona y asíncrona. El USART puede configurarse de modo asíncrono full dúplex el cual que puede comunicar dispositivos periféricos como ordenadores personales, o como un sistema síncrono half duplex para comunicarse con otros microcontroladores, dispositivos periféricos como los convertidores A/D, los circuitos integrados D/A, etc, [33].

Registro de estado del transmisor

Este registro tiene la función de controlar la transmisión serial y su estado, se conforma de 8 bits y sus funciones van desde elegir el tipo de transmisión hasta la velocidad de transmisión. La función de cada bit se describe a continuación:

TXSTA

CSRC	TX9	TXEN	SYNC	-	BRGH	TRMT	TX9D

bit 7: **CSRC**: Selecciona la fuente de reloj para transmisión síncrona.
bit 6: **TX9**: Habilita la transmisión serial a 9 bits.
bit 5: **TXEN**: Habilita la transmisión serial.
bit 4: **SYNC**: Selecciona el tipo de transmisión.
1: Síncrona
0: Asíncrona
bit 3: No esta implementado.
bit 2: **BRGH**: Selecciona la velocidad de tasa de baudios para transmitir en modo asíncrono.
bit 1: **TRMT**: Indica cuando el registro de desplazamiento del transmisor (TSR) esta vacio.
bit 0: **TX9D**: Noveno bit de datos o puede usarse como bit de paridad.

Registro de estado del receptor

Consta de 8 bits cuya función es configurar el estado y el control del receptor en la transmisión serial. Las funciones de cada bit se describen a continuación:

RCSTA

SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D

bit 7: **SPEN**: Habilita puerto de comunicación serial (configura los pines RC7/RX/DT y RC6/TX/CK para el puerto serie).
bit 6: **RX9**: Habilita la recepción serial a 9 bits.
bit 5: **SREN**: Habilita la recepción única en el modo síncrono maestro, en modo asíncrono no importa.
bit 4: **CREN**: Habilita recepción continua para modo síncrono y asíncrono.
bit 3: **ADDEN**: Habilita la detección de dirección.
bit 2: **FERR**: Indica error de empaquetamiento.
bit 1: **OERR**: Indica error por desbordamiento.
bit 0: **RX9D**: Noveno bit de datos de recepción o puede usarse como bit de paridad.

2.2.3. Pantalla de cristal líquido (LCD)

Es un módulo microcontrolado capaz de representar 2 líneas de 16 caracteres cada una (Figura 2.7). A través de 8 ó 4 líneas de datos (se elige mediante programación), se envía el carácter ASCII que se desea visualizar, así como ciertos códigos de control que permiten realizar diferentes efectos de visualización. Con otras tres señales adicionales se controla el flujo de información entre el módulo LCD y el dispositivo de control, [34].

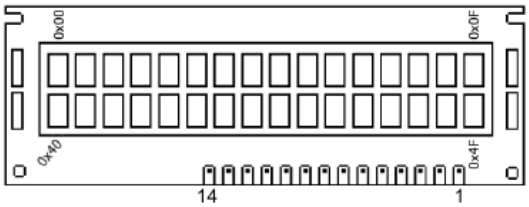

Figura 2.7: LCD de 2 líneas x 16 caracteres.

En la Tabla 2.4 se da la descripción de señales empleadas por el módulo LCD así como el número de pin correspondiente, [34].

Terminal	**Símbolo**	**Función**
1	Vss	Tierra ó Masa
2	Vdd	Alimentación + 5 VDC
3	Vo	Voltaje de ajuste de contraste
4	RS	Selección de Dato / Comando
5	R/W	Lectura / Escritura
6	E	Habilitador
7-14	D0-D7	Bus de datos

Tabla 2.4: Descripción de terminales del módulo LCD.

Para el control del LCD se dispone del puerto correspondiente en el sistema mínimo que corresponde a los dos primeros pines del puerto B (RS,E), y la parte alta del mismo puerto (D4-D7), debido a que se utiliza el formato de envío de datos a 4 líneas desde el microcontrolador.

Tabla de instrucciones

Para el funcionamiento del LCD, se dispone de una serie de instrucciones o palabras de control que se forman con una palabra de 8 bits (provenientes del bus de datos, D0-D7) + 2 bits (que se obtienen de los pines RS y R/W). Estas instrucciones se muestran en la Figura 2.8, [35].

Instruction	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	Description	Clock-Cycles
NOP	0	0	0	0	0	0	0	0	0	0	No Operation	0
Clear Display	0	0	0	0	0	0	0	0	0	1	Clear display & set address counter to zero	165
Cursor Home	0	0	0	0	0	0	0	0	1	x	Set adress counter to zero, return shifted display to original position. DD RAM contents remains unchanged.	3
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Set cursor move direction (I/D) and specify automatic display shift (S).	3
Display Control	0	0	0	0	0	0	1	D	C	B	Turn display (D), cursor on/off (C), and cursor blinking (B).	3
Cursor / Display shift	0	0	0	0	0	1	S/C	R/L	x	x	Shift display or move cursor (S/C) and specify direction (R/L).	3
Function Set	0	0	0	0	1	DL	N	F	x	x	Set interface data width (DL), number of display lines (N) and character font (F).	3
Set CGRAM Address	0	0	0	1	CGRAM Address						Set CGRAM address. CGRAM data is sent afterwards.	3
Set DDRAM Address	0	0	1	DDRAM Address							Set DDRAM address. DDRAM data is sent afterwards.	3
Busy Flag & Address	0	1	BF	Address Counter							Read busy flag (BF) and address counter	0
Write Data	1	0	Data								Write data into DDRAM or CGRAM	3
Read Data	1	1	Data								Read data from DDRAM or CGRAM	3

x : Don't care						
	I/D	1 0	Increment Decrement	R/L	1 0	Shift to the right Shift to the left
	S	1 0	Automatic display shift	DL	1 0	8 bit interface 4 bit interface
	D	1 0	Display ON Display OFF	N	1 0	2 lines 1 line
	C	1 0	Cursor ON Cursor OFF	F	1 0	5x10 dots 5x7 dots
	B	1 0	Cursor blinking	DDRAM : Display Data RAM CGRAM : Character Generator RAM		
	S/C	1 0	Display shift Cursor move			

Figura 2.8: Instrucciones para manejo de LCD.

2.2.4. Interfaz de comunicación serial RS-232

Las transmisiones serie se caracterizan por utilizar una única línea para transmitir la información. Esto obliga en el transmisor a convertir los bytes de paralelo a serie para enviar los bits de forma secuencial, por su parte en el receptor, la serie de bits recibidos son convertidos de serie a paralelo para reconstruir el byte. Inicialmente el puerto serie se utilizó para conectar la PC y un módem. En esta conexión, a la PC se le denomina DTE (Data Terminal Equipment) y al Módem DCE (Data Communication Equipment).

Existen dos tipos posibles de transmisiones serie: síncrona y asíncrona. En las transmisiones asíncronas a cada byte de información a transmitir se le añaden una serie de bits fijos de señalización para marcar el comienzo (bit de arranque o start) y el final de cada byte (bits de parada o stop).

El DTE puede ser un dispositivo inteligente como una PC, mientras que el DCE

suele ser un dispositivo no inteligente como un módem, una impresora, etc. Por lo que se estableció la norma RS-232, la cual define una velocidad de transmisión serial de hasta 20 Kbps y longitud máxima de 15 metros; y las señales eléctricas se definen entre +3 y +15 volts para el estado activo (uno lógico) y de -3 y -15 para el estado inactivo (cero lógico), además de que solo permite conectar dos dispositivos para establecer comunicación punto a punto.

Actualmente la mayoría de las PC poseen dos puertos serie definidos como COM1 y COM2 a nivel de sistema operativo y utilizan conectores DB9 macho. La ventaja de esta técnica es que es más fácil de implementar, pero una de sus desventajas es que además de cada byte de información se envían tres bits más que no son de información.

Para este proyecto se utiliza el registro USART en su modo asíncrono, comúnmente usado para comunicar al puerto serial usando el protocolo RS-232 por lo que se requirió una interfaz a RS-232 debido a los distintos niveles de voltaje, los cuales no deben ser conectados directamente a las señales RS-232.

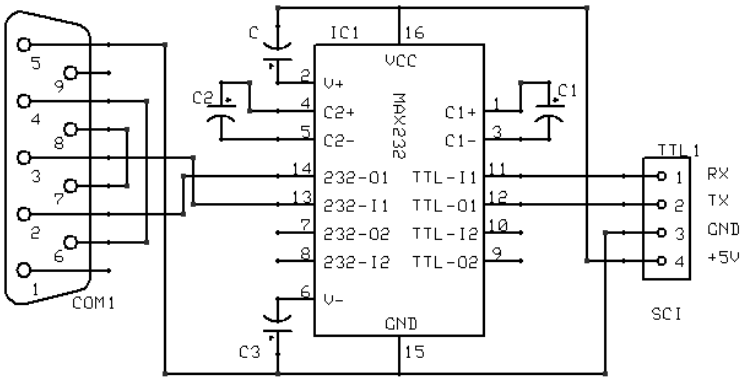

Figura 2.9: Diagrama esquemático de la interfaz RS-232.

La interfaz utiliza un circuito integrado MAX232 (Figura 2.9), el cual requiere de muy pocos componentes externos para adecuar los voltajes de tipo digital al protocolo RS-232 y la fuente de alimentación puede ser la misma que la del sistema mínimo (+5VCD),y se puede conectar directamente con cualquier puerto serial de la computadora (COM1 o COM2).

2.2.5. Sensores

El *sensor* es el dispositivo que convierte un tipo de energía en otro tipo de energía, generalmente una señal eléctrica. Pero para su conversión A/D se requiere de acondicionadores de señal para adecuar correctamente los niveles de voltaje. Todos los acondicionadores estarán ajustados a las características eléctricas del CAD del PIC16F877

(0-5VCD), y deberán tener una respuesta lineal a la salida. Además en todos ellos la señal de salida del CAS (circuito acondicionador de señal) deberá ser menor o igual a un volt cuando la variable climática esté en el mínimo de su especificación de diseño, esto con el fin de sujetar la señal a un nivel de voltaje para advertir de mal funcionamiento en algún circuito [36].

A continuación se muestran los requerimientos básicos para el diseño de los distintos acondicionadores de señales (para detalles sobre el diseño de los acondicionadores de señal consultar la referencia bibliográfica [36]). La salida de cada CAS es aplicada a un pin del microcontrolador configurada como entrada analógica en el caso de temperatura y velocidad de viento.

Temperatura

El margen de temperatura que se va a medir está comprendido entre -5 a 50°C. Cuando la temperatura medida por el sensor es de -5°C, la salida del CAS será de 1 V; cuando el sensor mida 50°C la salida deberá ser de 5V [36].

El circuito integrado LM35 ó LM35A es el sensor de temperatura que se utiliza y entre sus principales características esta el que su tensión de salida es linealmente proporcional con la temperatura en la escala Celsius (grados centígrados).El circuito posee una precisión aceptable para la aplicación requerida, opera con alimentaciones que van desde 4 a 20V, no necesita calibración externa, posee sólo tres terminales, permite la medición remota y es de bajo costo.

Velocidad de viento

El intervalo de Velocidad de viento (VV) que se va a medir esta comprendido entre 0 y 90Km/h. Cuando la VV medida por el anemómetro es de 0Km/h, la salida del CAS será de 1 V; cuando el sensor mida 90Km/h la salida deberá ser de 5V, [36].

Para medir la velocidad del viento se utiliza un anemómetro rotativo de cubetas, que utiliza un transductor electromagnético, el cual entrega una corriente proporcional a la velocidad del viento. Los detalles del diseño se pueden consultar en [36].

Presencia de lluvia

Este sensor deberá entrega una señal de salida digital para indicar solo la ausencia o presencia de lluvia, ya que para la determinación de los rangos no es muy relevante el que exista poca o mucha lluvia, simplemente se considera como 0 ó 1 (ausencia o

presencia respectivamente).
Esta señal será leída desde un pin configurado como entrada digital en el microcontrolador, por lo que no debe ser mayor a 5VCD para no dañar al dispositivo.

Posición de cortinas

Para sensar los niveles de apertura de las cortinas se emplean interruptores normalmente abiertos llamados también llaves magnéticas o reed-switch, los cuales realimentan al sistema de control el estado de las cortinas. Existen cuatro estados posibles: cerrada, abierta completamente, 33 % abierta y 66 % abierta.

Los reed-switch son unas pequeñas cápsulas de vidrio que encierra un par de contactos metálicos con conexión al exterior. En presencia de un campo magnético apropiado, los contactos dentro de la cápsula se tocan, cerrando el interruptor. En la Figura 2.10 se muestra el sensor, y el circuito a implementar para cada interruptor se muestra en la Figura 2.11. Los sensores de cada posición se conectan a un pin del microcontrolador configurado como entrada digital en el puerto D.

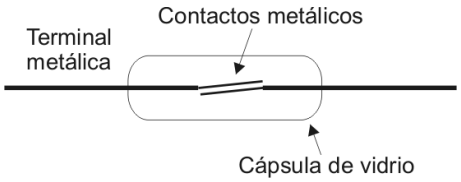

Figura 2.10: Reed-switch.

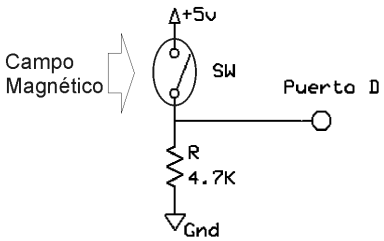

Figura 2.11: Circuito básico de conexión para el reed-switch.

El imán que activa los interruptores deberá ir montado sobre la parte móvil de la estructura y el reed-switch en una parte fija de la estructura (Figura 2.12). A modo de darle mayor confiabilidad al sistema, se puede colocar en forma duplicada los sensores magnéticos en cada una de las posiciones de apertura de las cortinas.

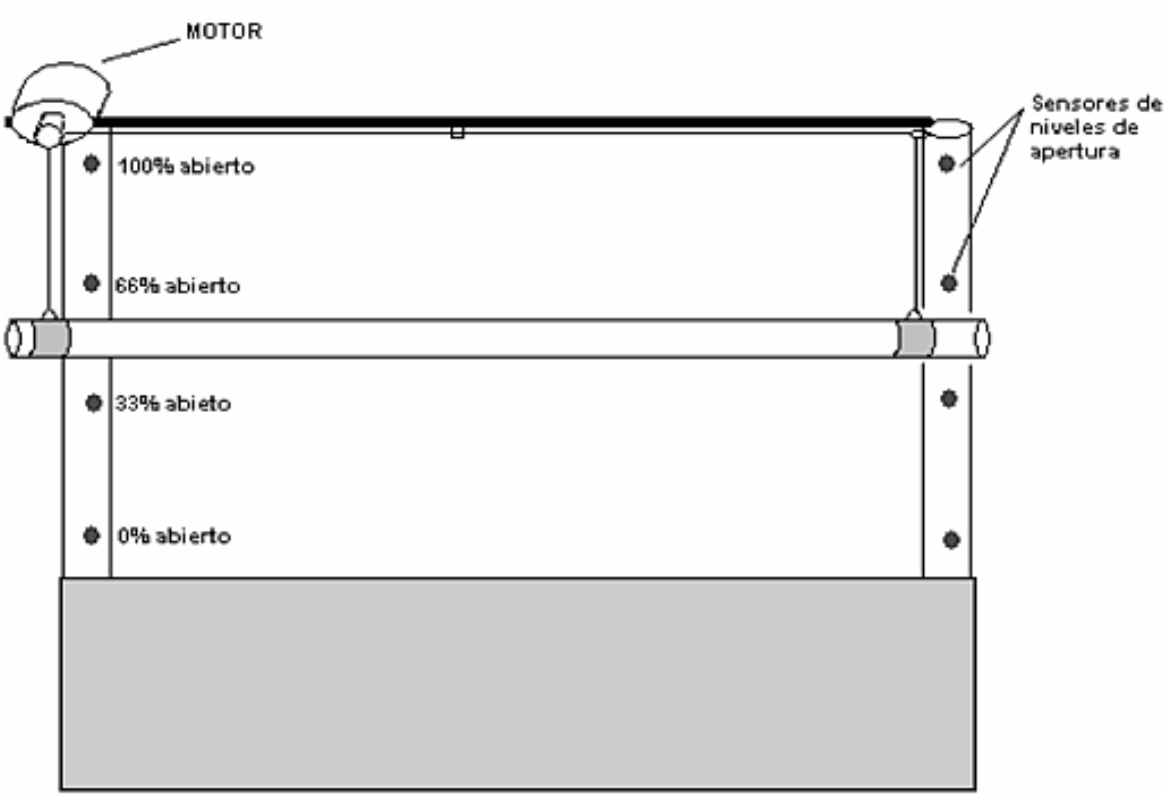

Figura 2.12: Esquema de cortinas para montar los sensores de posición.

Capítulo 3

Desarrollo del autómata

En este capítulo se describen las consideraciones tomadas en cuenta para el diseño del autómata en el cual se basa el algoritmo de control para la apertura y cierre de las cortinas del invernadero.

Como se mencionó en el Capítulo 1, la funcionalidad de un invernadero radica principalmente en proveer un microclima adecuado a la plantación. Por lo tanto la variable a controlar en este sistema es la temperatura interior. Las variables independientes que inciden significativamente en la dinámica de esta variable son: temperatura exterior, velocidad del viento y presencia de lluvia.

Existen invernaderos con sistemas que son equipados con reguladores independientes para controlar la calefacción, humedad en el interior, cortinas, etc. Tales sistemas de regulación a menudo son insuficientes porque no toman en cuenta consideraciones relacionadas entre las diferentes variables, [37].

Por consiguiente se pretende que todas estas variables incidan en la temperatura interior de tal modo que esta se mantenga en un rango entre 20°C y 30°C.

Antes de comenzar a describir el autómata para el control de las cortinas, es necesario conocer conceptos básicos sobre Autómatas Finitos, los cuales se verán en la siguiente sección.

3.1. Autómatas finitos

Los autómatas finitos podrían definirse como maquinas abstractas simples, las cuales representan solamente el aspecto referente a las secuencias o ciclos de acciones en un sistema. Existen muchas situaciones de la realidad que pueden ser modeladas usando dichos autómatas, [38].

3.1.1. Modelado de eventos discretos

Los eventos discretos son aquellos en los que se considera su estado solo en ciertos momentos, separados por intervalos de tiempo, sin importar lo que ocurre en el sistema entre estos momentos. Es como si la evolución del sistema fuera descrita por una secuencia de fotografías, en vez de un flujo continuo, y se pasa bruscamente de una fotografía a otra.

Generalmente se considera que la realidad es continua, y por lo tanto los sistemas discretos son solamente una abstracción de la realidad.

La noción mas básica de los modelos de eventos discretos es la de *estado*. Un estado es una situación en la que se permanece un cierto lapso de tiempo y, de uno de estos estados se puede pasar a otro al ocurrir un evento o acción. En estos modelos se supone que se permanece en los estados un cierto tiempo, pero por el contrario, los eventos son instantáneos, [38], [39].

Resulta práctico expresar los modelos de estados y eventos de manera gráfica. Los estados se representan por círculos, y los eventos por flechas entre los círculos, llamadas *transiciones*. Además, las secuencias de eventos pueden representarse por concatenaciones de caracteres, esto es, por *palabras*. Dentro de cada estado se escribe su nombre, mientras que al lado de las transiciones se escribe el nombre del evento asociado, como en la Figura 3.1. El estado inicial se indica con '▷', [38].

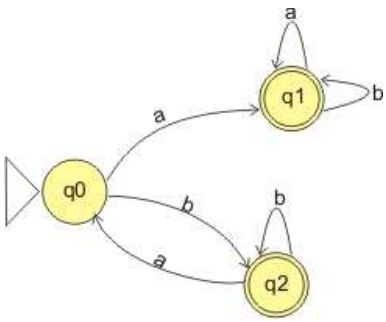

Figura 3.1: Notación gráfica de un autómata finito.

Para elaborar modelos adecuados de un proceso real [38], se deben considerar los siguientes puntos:

1. Diferenciar entre los eventos que se consideran instantáneos y aquellos que tienen una duración considerable: estos últimos se asocian a los estados. Los estados son la base de un diseño de los modelos que están estudiando, pues representan las situaciones básicas por las que pasa el proceso.

2. Las condiciones asociadas a los estados deben ser excluyentes, esto es, no deben verificarse varias simultáneamente.

3. Las condiciones asociadas a los estados de un modelo bien hecho deben ser comprensivas, lo que quiere decir que entre todas ellas cubren todos los casos posibles.

4. Los eventos instantáneos son asociados a los eventos o transiciones.

Estados finales

El propósito de algunos modelos de estados y eventos es el de reconocer secuencias de eventos adecuadas, de manera que se les pueda diferenciar de las secuencias inadecuadas.

Un aspecto muy importante del modelo de la Figura 3.1 es que los estados q_1 y q_2 son estados especiales, llamados estados finales y son identificados por un circulo con doble trazo. Los estados finales indican que cuando se llega a ellos, la secuencia de eventos que llevó hasta ahí puede considerarse como *aceptable*, [38], [39].

3.1.2. Funcionamiento de los autómatas finitos

El funcionamiento de los autómatas finitos consiste en ir pasando de un estado a otro, a medida que va recibiendo los caracteres de la palabra de entrada (transiciones). Este proceso puede ser seguido fácilmente en los diagramas de estados. Simplemente hay que pasar de estado a estado siguiendo las flechas de las transiciones, para cada caracter de la palabra de entrada, empezando por el estado inicial.

Por ejemplo, supongase que se tiene el autómata de la Figura 3.1 y la palabra de entrada bb. El autómata inicia su operación en el estado q_0 (que es el estado inicial), y al recibir la primera b pasa al estado q_2, pues en el diagrama hay una flecha de q_0 a q_2 con la letra b. Luego, al recibir la segunda b de la palabra de entrada, pasara del estado q_2 a él mismo, pues en la figura se puede ver una flecha que de q_2 regresa al mismo estado, con la letra b.

Podemos visualizar el camino recorrido en el diagrama de estados como una *trayectoria* recorrida de estado en estado. Por ejemplo, para el autómata finito de la Figura 3.1 la trayectoria seguida para la palabra ab consiste en la secuencia de estados: q_0, q_1, q_1.

Los estados son el único medio de que disponen los AF para recordar los eventos que ocurren (por ejemplo, que caracteres se han leído hasta el momento); esto quiere decir que son maquinas de memoria limitada, [38].

3.1.3. Definición formal de autómatas finitos

Al describir una maquina de estados finitos en particular, debemos incluir las informaciones que varían de un autómata a otro, es decir, no tiene sentido incluir descripciones generales aplicables a todo autómata. Estas informaciones son exactamente las que aparecen en un diagrama de estados y transiciones, como el de la Figura 3.1.

A continuación se presenta un formato matemático para representar las mismas informaciones que contiene un diagrama de estados. Como se utiliza terminología matemática en vez de dibujos, se dice que se trata de una notación formal, [37], [38], [39].

Definición.- Una maquina de estados finitos M es un quíntuplo formado por:

$$M = (Q, \Sigma, \delta, s, F), \tag{3.1}$$

donde

- Q es un conjunto de identificadores (símbolos) de estados;
- Σ es el alfabeto de entrada;
- $s \in Q$ es el estado inicial;
- $F \subseteq Q$ es un conjunto de estados finales;
- $\delta : Q \text{ x } \Sigma \rightarrow Q$ es la función de transición, que a partir de un estado y un símbolo del alfabeto obtiene un nuevo estado.

La función de transición indica a que estado se va a pasar sabiendo cuál es el estado actual y el símbolo que se esta leyendo. *Es importante notar que δ es una función y no simplemente una relación; esto implica que para un estado y un símbolo del alfabeto dados, habrá un y solo un estado siguiente.* Esta característica, que permite saber siempre cual sera el siguiente estado, se llama determinismo. La definición dada arriba corresponde a los autómatas finitos deterministas, abreviado AFD, [38].

Ejemplo.- El autómata finito determinista de la Figura 3.1 puede ser expresado formalmente como: $M = (Q, \Sigma, \delta, q_0, F)$, donde

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{a, b\}$
- $\delta = \{((q_0, a), q_1), ((q_0, b), q_2), ((q_1, a), q_1), ((q_1, b), q_1), ((q_2, a), q_0), ((q_2, b), q_2)\}$
- $F = \{q_1, q_2\}$

La función de transición δ puede ser expresada mediante la Tabla 3.1 en la cual las filas representan cada uno de los estados del autómata y, las columnas representan cada uno de los caracteres de entrada . Para saber a qué estado pasa el autómata, basta con ubicar el estado actual (q) y el caracter de entrada (σ, para así obtener el estado al cual pasa (q') en la intersección.

	a	b
q_0	q_1	q_2
q_1	q_1	q_1
q_2	q_0	q_2

Tabla 3.1: Representación de la función de transición.

Es fácil ver que la diferencia entre los diagramas de estado y los AFD en notación formal es solamente de notación, siendo la información exactamente la misma, por lo que es sencillo pasar de una representación a la otra.

Tanto en los diagramas de estado como en la representación formal hay que tener cuidado en respetar las condiciones para tener un autómata valido; en particular, el numero de transiciones que salen de cada estado debe ser igual a la cantidad de caracteres del alfabeto, puesto que δ es una función que esta definida para todas las entradas posibles, [38].

Para el ejemplo de la Figura 3.1, donde el alfabeto es $\{a,\ b\}$, de cada estado deben salir exactamente dos transiciones, una con a y otra con b.

Otra condición es que debe haber exactamente un estado inicial. En cambio, la cantidad de estados finales puede ser cualquiera, inclusive cero, hasta un máximo de |Q| (la cantidad de estados).

En la notación formal también hay que seguir las transiciones, que ahora no son representadas como flechas, sino como elementos del conjunto δ de transiciones. Tomando nuevamente el autómata de la Figura 3.1 y la palabra de entrada bb, la operación se inicia en el estado inicial q_0; luego, al recibir la primera b, usando la transición $((q_0, b), q_2)$ pasa a q_2, y luego, al recibir la segunda b de la palabra de entrada, por medio de la

transición $((q_2, b), q_2)$ pasa al estado q_2 (de hecho permanece en el).

De una manera mas general, si un AFD se encuentra en un estado q y recibe un caracter σ pasa al estado q' sí y sólo sí

$$\delta(q, \sigma) = q', \tag{3.2}$$

esto es, si $((q, \sigma), q') \in \delta$, [38].

3.2. Descripción general del autómata

Para este proyecto se consideran tres variables, las cuales inciden de manera significativa en el control de apertura de las cortinas laterales del invernadero: Temperatura Exterior, Velocidad de Viento y Presencia de Lluvia, las cuales se toman como variables de entrada, y como variable de salida se toma la apertura de las cortinas.

Es importante mencionar que, la razón por la cual se decide trabajar con los datos de temperatura exterior, es porque al considerar las mediciones de la temperatura interior para el controlador de las cortinas, se ocasionaría un conflicto con el controlador de la calefacción, lo que a su vez significaría perdidas de energía; por ello se muestra el análisis de la relación entre el comportamiento de la temperatura exterior y la temperatura interior en los mismos periodos de tiempo(Figura 3.2), para así poder proponer los rangos adecuados en los que las cortinas se deben accionar con respecto a la temperatura exterior. Las gráficas mostradas corresponden al registro de las variables de un invernadero ubicado en la Universidad del Sur de Toulon Var, Francia.

Como se puede apreciar en la Figura 3.2, cuando la temperatura exterior es menor a 12°C, la temperatura interior es menor a 25°C, por lo que las cortinas se mantiene cerradas debido a que la temperatura que favorece el desarrollo de las plantas se encuentra alrededor de 24°C. Por ejemplo, para mantener las cortinas completamente abiertas tomamos un valor de temperatura exterior mayor a 18°C, ya que por encima de ese valor la temperatura interior es mayor a 30°C, lo que afectaría al cultivo.

En la Figura 3.3 se muestra el rango para la temperatura interior que se considera favorable para el cultivo de las plantas. Por debajo de ese rango se activa la calefacción dentro del invernadero para subir la temperatura (azul). Por encima de ese rango se deberán abrir por niveles las cortinas de ventilación para disminuir la temperatura (rojo).

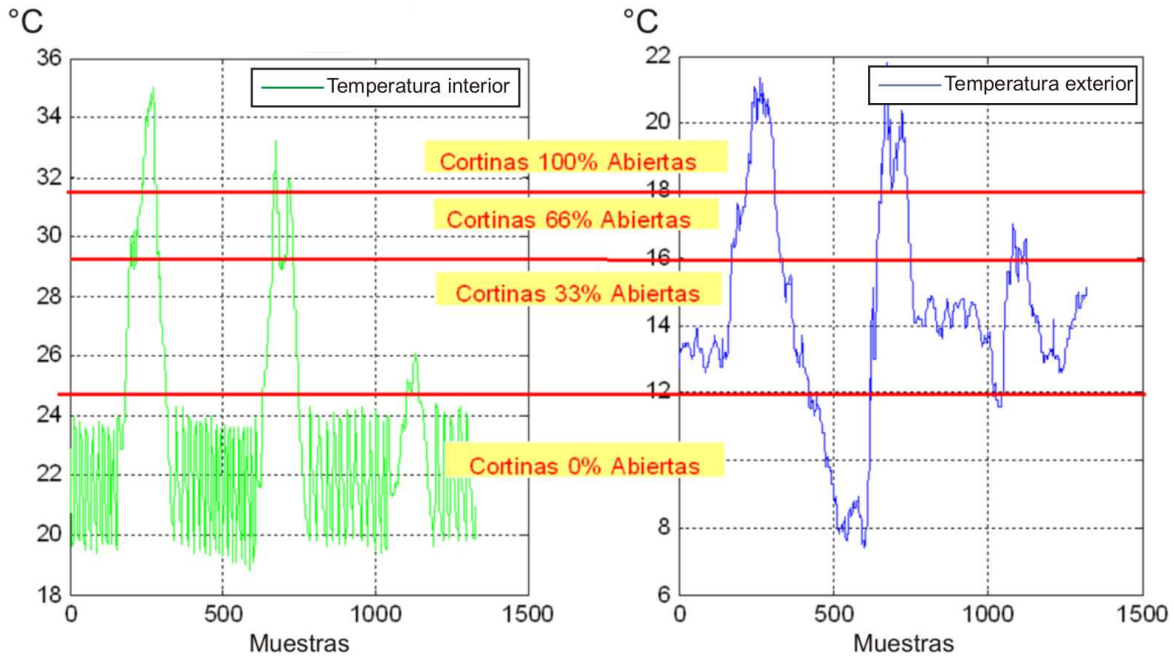

Figura 3.2: Relación entre temperatura interior y exterior.

Para el diseño del autómata se toman en cuenta las siguientes condiciones:

- Las cortinas se mantendrán en la posición 1 (0 % abierto), mientras $lluvia = 1$, ó la temperatura exterior $\leq 12^oC$, ó velocidad viento $> 15Km/h$.
- Cuando $lluvia = 0$, las cortinas se mantendrán en la posición 2 (33 % abierto), mientras la temperatura exterior $> 12^oC$ y temperatura exterior $\leq 16^oC$, ó velocidad viento $> 10Km/h$ y velocidad viento $\leq 15Km/h$, ó .
- Cuando $lluvia = 0$, las cortinas se mantendrán en la posición 3 (66 % abierto), mientras la temperatura exterior $> 16^oC$ y temperatura exterior $\leq 18^oC$, ó velocidad viento $> 5Km/h$ y velocidad viento $\leq 10Km/h$.
- Cuando $lluvia = 0$, las cortinas se mantendrán en la posición 4 (100 % abierto), mientras la temperatura exterior $> 18^oC$, ó velocidad viento $\leq 5Km/h$.

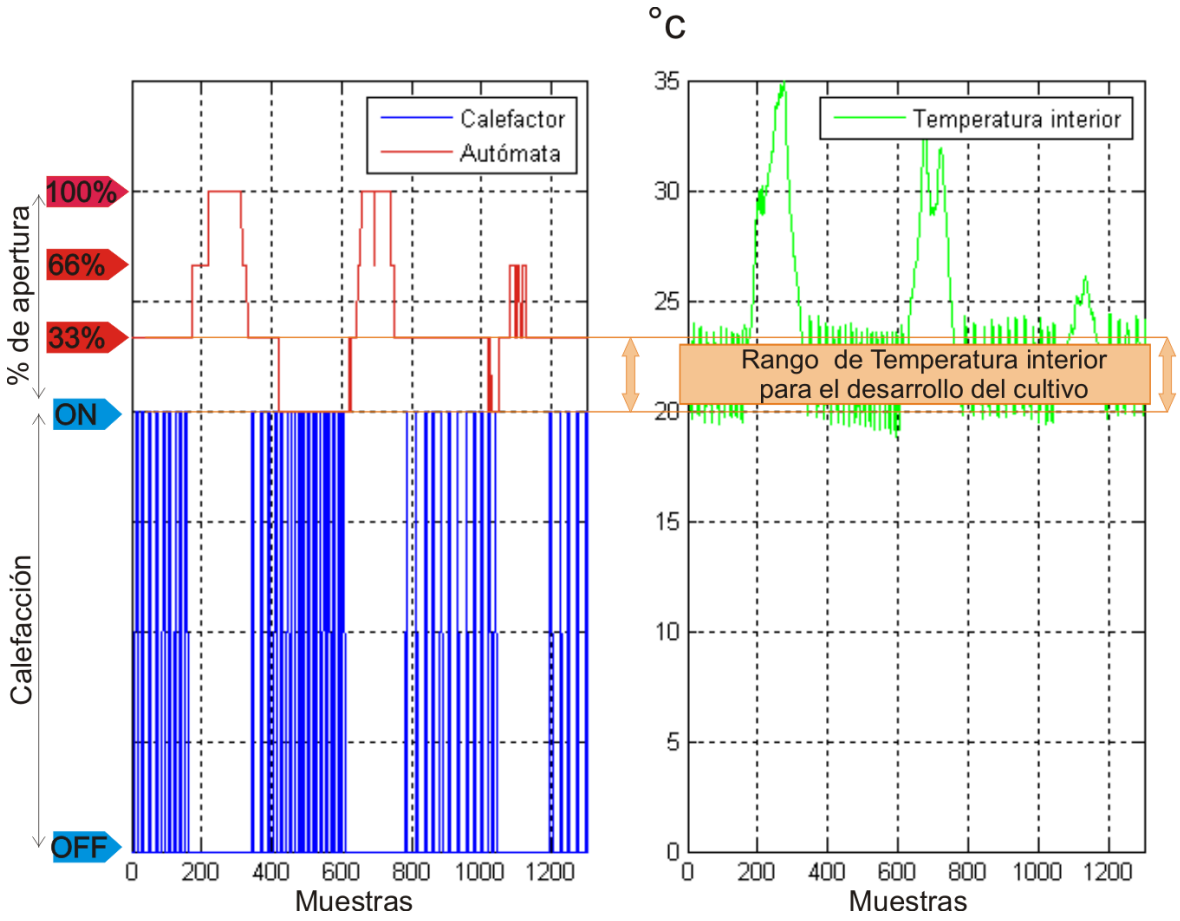

Figura 3.3: Acción del calefactor y del Autómata.

3.2.1. Representación del autómata

Para facilitar la representación del autómata, tanto de manera gráfica, como de manera formal, a cada estado se le asigna un caracter con un subíndice y la descripción de cada uno de ellos se muestra en la Tabla 3.2.

En la Tabla 3.3 se muestra la descripción de cada una de las transiciones del sistema y los caracteres asignados a cada una de ellas.

El diagrama de transiciones para el autómata propuesto se creó en ambiente Jflap (Sección 2.1.3) y se muestra en la Figura 3.4. Como ya se mencionó, los círculos representan a cada estado y las flechas representan la transición de un estado a otro.

Estado	Descripción
q_0	Estado inicial
q_1	Comparando posición 1
q_2	Comparando posición 2
q_3	Comparando posición 3
q_4	Comparando posición 4
q_5	Cerrando hasta 0 % abierto
q_6	Cerrando hasta 33 % abierto
q_7	Cerrando hasta 66 % abierto
q_8	Abriendo hasta 33 % abierto
q_9	Abriendo hasta 66 % abierto
q_{10}	Abriendo hasta 100 % abierto
q_{11}	0 % abierto
q_{12}	33 % abierto
q_{13}	66 % abierto
q_{14}	100 % abierto

Tabla 3.2: Descripción de los estados del autómata.

Transición	Descripción
$Ll = 0$	Ausencia de lluvia
$Ll = 1$	Presencia de lluvia
$T1$	Temperatura exterior $\leq 12^oC$
$T2$	Temperatura exterior $> 12^oC$ y $\leq 16^oC$
$T3$	Temperatura exterior $> 16^oC$ y $\leq 18^oC$
$T4$	Temperatura exterior $> 18^oC$
$V1$	Velocidad de viento $> 15Km/h$
$V2$	Velocidad de viento $> 10Km/h$ y $\leq 15Km/h$
$V3$	Velocidad de viento $> 5Km/h$ y $\leq 10Km/h$
$V4$	Velocidad de viento $\leq 5Km/h$
$P1$	Posición 1
$P2$	Posición 2
$P3$	Posición 3
$P4$	Posición 4

Tabla 3.3: Descripción de las transiciones del autómata.

En el diagrama de estados del autómata propuesto (Figura 3.4), se resalta una de las trayectorias en color rojo, la cual se describe a manera de ejemplo y señala una de las secuencias para la lectura de las variables.

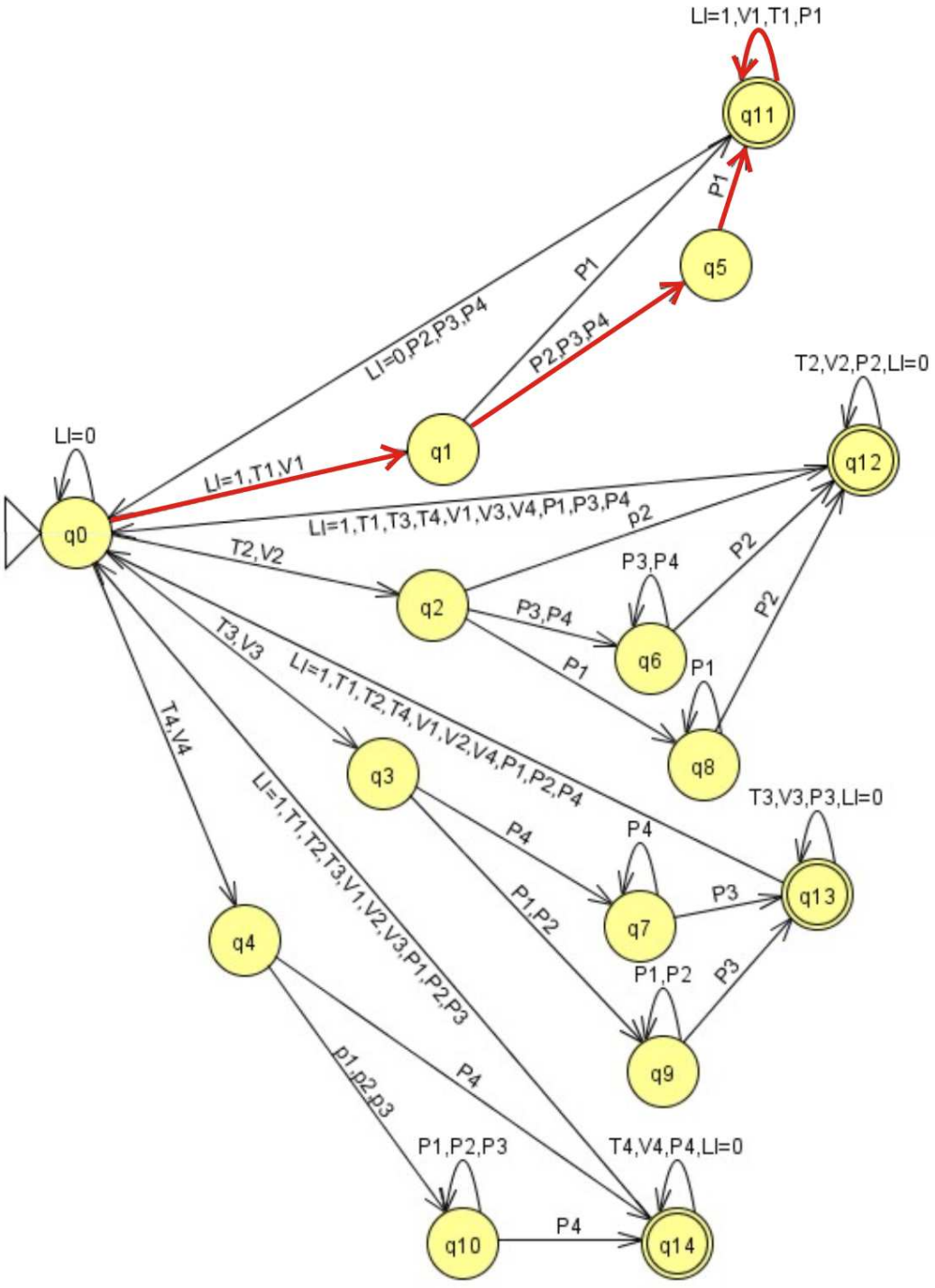

Figura 3.4: Diagrama del AFD.

Supongase que se tienen las siguientes condiciones:

- Presencia de lluvia ($Ll = 1$).
- Posición de las cortinas 33 % abierto($P2$).
- Temperatura exterior de 8^oC ($T1$ de acuerdo a la Tabla 3.3).
- Velocidad de viento de $20Km/h$ ($V1$ de acuerdo a la Tabla 3.3).

De acuerdo a estas condiciones, (variables de entrada o transiciones), el automata inicia su operación a partir del estado inicial q_0, al existir presencia de lluvia pasa a q_1, pues hay una flecha de q_0 a q_1 con $Ll = 1$; de q_1 pasa a q_5 debido a que la posición que lee el autómata es $P2$; mientras la posición que lee es diferente de $P1$, permanece en q_5 y pasa a q_{11} al leer $P1$; el autómata permanece en q_{11} mientras siga leyendo $Ll = 1$ ó $V1$ ó $T1$ ó $P1$.

Representación formal del autómata

De acuerdo a la Ecuación 3.1, el autómata finito determinista propuesto se representa en notación formal como:

- $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_{12}, q_{13}, q_{14}\}$
- $\Sigma = \{Ll = 0, Ll = 1, T1, T2, T3, T4, V1, V2, V3, V4, P1, P2, P3, P4\}$
- $s = q_0$
- $F = \{q_{11}, q_{12}, q_{13}, q_{14}\}$

La función de transición (δ) para el autómata se representa con la Tabla 3.4, donde se muestra el estado al que pasa el autómata para un estado y una transición dada (Ecuación 3.2). Cada fila representa a cada uno de los 15 estados y cada columna representa una transición o caracter de entrada (en este caso, en el autómata se tienen 14 transiciones), la letra griega TETA (θ) representa un estado vacío.

	$Ll=0$	$Ll=1$	$T1$	$T2$	$T3$	$T4$	$V1$	$V2$	$V3$	$V4$	$P1$	$P2$	$P3$	$P4$
q_0	q_0	q_1	q_1	q_2	q_3	q_4	q_1	q_2	q_3	q_4	θ	θ	θ	θ
q_1	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_{11}	q_5	q_5	q_5
q_2	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_8	q_{12}	q_6	q_6
q_3	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_9	q_9	q_{13}	q_7
q_4	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_{10}	q_{10}	q_{10}	q_{14}
q_5	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_{11}	q_5	q_5	q_5
q_6	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_{12}	q_6	q_6
q_7	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_{13}	q_7
q_8	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_8	q_{12}	θ	θ
q_9	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_9	q_9	q_{13}	θ
q_{10}	θ	θ	θ	θ	θ	θ	θ	θ	θ	θ	q_{10}	q_{10}	q_{10}	q_{14}
q_{11}	q_0	q_{11}	q_{11}	θ	θ	θ	q_{11}	θ	θ	θ	q_{11}	q_0	q_0	q_0
q_{12}	q_{12}	q_0	q_0	q_{12}	q_0	q_0	q_0	q_{12}	q_0	q_0	q_0	q_{12}	q_0	q_0
q_{13}	q_{13}	q_0	q_0	q_0	q_{13}	q_0	q_0	q_0	q_{13}	q_0	q_0	q_0	q_{13}	q_0
q_{14}	q_{14}	q_0	q_0	q_0	q_0	q_{14}	q_0	q_0	q_0	q_{14}	q_0	q_0	q_0	q_{14}

Tabla 3.4: Representación de la función de transición del autómata.

3.3. Descripción del modelo en MATLAB

Mediante MATLAB se obtuvieron las simulaciones del automata finito determinista de acuerdo a las variables consideradas en su diseño (Capítulo 4).

La base de datos contiene las mediciones obtenidas en un muestreo tomado durante tres días, de los cuales se obtuvieron 1329 datos, es decir, un dato por cada 4 minutos.

A continuación se muestra el código del programa en ambiente MATLAB. Los comentarios se indican con un signo de %:

```
       % automata
 clc;clf;clear all;

       % Se carga la base de datos '10_11_12jan'
       % para activar las variables siguientes
load 10_11_12jan
ov=data(:,1)';          %cortinas de ventilación
ch=data(:,2)';          %calefactor
te=data(:,3)';          %temperatura exterior
he=data(:,4)';          %humedad exterior
ray=data(:,5)';         %radiación solar
vv=data(:,6)';          %velocidad del viento
ti=data(:,7)';          %temperatura interior
hi=data(:,8)';          %humedad interior
bru=data(:,9)';         %sistema de brumización
om=data(:,10)';         %cobija térmica
ll=data(:,11)';         %lluvia se crea y guarda n la misma base de datos
atm=data(:,12)';        %automata se crea y guarda n la misma base de datos

       %iteraciones para calcular valor de lluvia y cierre de cortinas
n=length(te);
for i=1:n                               % se lleva a cabo 1 iteración para cada muestra

if ((ray(i)<10)&(he(i)>65))             % condiciones para generar variable estimada de lluvia
    ll(i)=1;
else
    ll(i)=0;
end

       %% control de niveles de apertura de la cortina
if((te(i)<=12) | (vv(i)>15) | (ll(i)==1))                       % cierra totalmente
    atm(i)=0;
    elseif(((te(i)>12)&(te(i)<=16)) | ((vv(i)>10))&(vv(i)<=15)) % abre 33%
        atm(i)=33;
        elseif((te(i)>16)&(te(i)<=18)) | ((vv(i)>5)&(vv(i)<=10))% abre 66%
            atm(i)=66;
            elseif((te(i)>18) | (vv(i)<=5))                     % abre 100%
                atm(i)=100;
end
end                                     % terminan iteraciones
```

3.4. Desarrollo de la programación para el PIC16F877

El algoritmo de control de las cortinas de ventilación que se implementó en el sistema mínimo se creó en lenguaje C y con las herramientas que se describieron en el capítulo anterior, [27], [28]. Algunas librerías y funciones que se utilizan han sido modificadas de algunos ejemplos que se incluyen con el software de PICCLITE para adecuarlas al sistema mínimo utilizado. Existen muchas ventajas al programar en lenguaje C, por ejemplo: la declaración y uso de variables de cualquier tipo (entero, flotante, caracter, con signo, sin signo, etc.); el uso de librerías, que permite crear funciones que son llamadas desde un programa principal; el manejo de operaciones aritméticas.

3.5. Descripción del programa para el PIC16F877

El programa que hace posible el control de apertura para las cortinas del invernadero, y que es implementado en el sistema mínimo puede describirse en secciones de acuerdo a la función que realizan dentro de todo el programa. Estas secciones son:

- Lectura y actualización de variables.
- Cálculo de la variable posición siguiente.
- Despliegue de información en LCD.
- Activación del actuador.
- Envío de variables via puerto serie RS232.

3.5.1. Lectura y actualización de variables

Esta parte del programa consiste en la conversión A/D y el cálculo de las variables climatológicas (temperatura y viento) y del registro de las variables de tipo digital (presencia de lluvia, posición de cortinas). De acuerdo a esta información se determina la posición en la cual debe mantenerse la cortina. También aquí se realiza el despliegue de todas las variables en LCD una vez que se han actualizado cada una de ellas

Para llevar a cabo lo anterior se crea la función *update*, la cual es llamada durante todo el programa por la importancia de conocer la posición de la cortina en todo momento, así como de las variables climatológicas.

Conversión analógico a digital

Sólo se hace la medición de dos canales analógicos, temperatura y velocidad de viento, para los cuales se utilizan variables de tipo caracter sin signo para guardar la parte alta del resultado de la conversión.

Lo primero que se debe hacer es configurar los puertos analógicos como entradas poniendo un *1* en el bit correspondiente del registro TRISA. Para seleccionar el canal 0 y el canal 1 basta con poner `0x03` en TRISA.

El registro ADCON1 se configura de acuerdo a la Sección 2.2.2 con `0x00`, con esto se justifica el resultado de la conversión a la izquierda (Figura 3.5), se seleccionan RA5, RA3:RA0, RE2:RE0, como canales analógicos y $V_{ref+} = VDD$ y $V_{ref-} = VSS$.

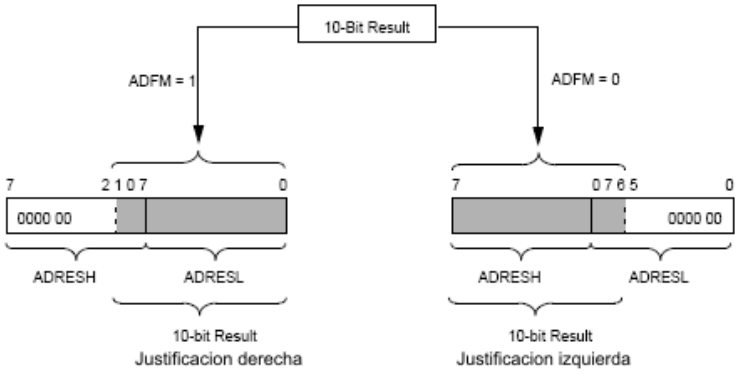

Figura 3.5: Justificación del resultado de la conversión A/D.

El registro ADCON0 se configura con `0x00` de acuerdo a la Sección 2.2.2. Así se tiene una frecuencia de reloj de conversión de $F_{osc}/2$ (Tabla 2.2). El canal 0 se selecciona y permanece apagado el módulo convertidor A/D.

Es importante que la conversión se inicie en una instrucción aparte de cuando el módulo convertidor se encienda y después de un pequeño retardo para evitar errores en la conversión o que no quede configurado correctamente. Esto se hace poniendo *1* en el bit ADGO (GO/DONE), del registro ADCON0. Al finalizar la conversión, el microcontrolador automáticamente coloca a *0* este bit.

El justificar a la izquierda el resultado de la conversión se hace con el fin de utilizar el registro ADRESH para obtener el valor de la variable que se está midiendo, ya que contiene los bits más significativos y es posible despreciar los bits menos significativos del resultado contenidos en el registro ADRESL, aunque se pierde un poco de resolución.

Al utilizar sólo el registro ADRESH del resultado de la conversión se tienen 255 niveles de cuantificación del voltaje máximo, que esta configurado como $V_{ref+} = VSS$. Para determinar el valor del voltaje a la entrada del convertidor se utiliza la Ecuación 3.3.

$$V_{in} = resolucion \cdot N_{bin} = \frac{V_{ref} \cdot ADRESH}{2^n - 1} \tag{3.3}$$

donde

- *resolución* (Ecuación 1.1), es el voltaje por cada incremento básico de la conversión A/D.

- N_{bin} es la cantidad en código binario obtenida del resultado de la conversión (ADRESH).

- n es la cantidad de bits del resultado de la conversión.

Para obtener el valor de la temperatura y la velocidad de viento se convierte el resultado de la conversión a voltaje y éste a su vez es modificado de acuerdo a la ecuación característica del CAS correspondiente, [36].

Para la Temperatura en ^oC, se usa la ecuación:

$$T = \frac{V_{in} - 1{,}363636364}{0{,}072727273} \tag{3.4}$$

Para la Velocidad de Viento en Km/h, se usa la ecuación:

$$VV = \frac{V_{in} - 1}{0{,}0444445} \tag{3.5}$$

Algoritmo de conversión A/D

El algoritmo de conversión es parte de la función *update* y consiste en :

- Seleccionar el canal analógico. Al ser dos canales analógicos los que se están utilizando, basta con modificar el bit CHS0 del registro ADCON0.

- Encender el módulo convertidor A/D. ADON = 1.

- Iniciar la conversión. (ADGO = 1).

- Esperar a que termine la conversión. Se puede hacer verificando el estado del bit ADGO.

- Guardar el resultado de la conversión. Se utilizan dos variables de 8 bits para guardar el valor de ADRESH. El canal 0 se guarda en *temp* y el canal 1 en *vien*.

- Obtener el valor de cada variable (Ecuaciones 3.4 y 3.5), y almacenarla como un dato de tipo flotante (*temperatura,viento*).

Posición de cortinas

En esta parte se manejan señales de tipo digital provenientes de los reed-switches ubicados en cada posición de la cortina (Figura 2.11). Para su conexión se utiliza la parte alta del Puerto D, que debe estar configurado como entradas en los bits correspondientes del registro TRISD. Con `0xff` se configura todo el Puerto D como entradas, aunque no se utilizan todas. A los pines utilizados se les designa de la siguiente manera:

1. RD4 = SW00 = posición 1
2. RD5 = SW33 = posición 2
3. RD6 = SW66 = posición 3
4. RD7 = SW100 = posición 4

Para modificar la posición en que se encuentra la cortina se verifica uno a uno cada pin y cuando hay *1* se actualiza la variable *posicion* con el valor correspondiente.

Presencia de lluvia

Se utiliza el pin RD3 del Puerto D configurado como entrada con el correspondiente bit en el registro TRISD como se vio en la sección anterior. A este pin se le designa como *lluvia* y sólo toma dos valores.

- 0 = Ausencia de lluvia.
- 1 = Presencia de lluvia.

Basta con leer el valor presente en ese bit en cualquier momento para tomar una decisión en el programa, que para este caso, es la de cerrar la cortina totalmente en presencia de lluvia.

3.5.2. Cálculo de la variable posición siguiente

En esta parte del programa principal se modifica el valor de la variable *sigpos* mediante condiciones *if else*, que evalúan los rangos establecidos de las variables Temperatura, Viento y Lluvia con el uso de operadores de comparación y operadores lógicos.

La variable *sigpos* toma los valores posibles de 1 a 4, de acuerdo con las siguientes consideraciones (Sección 3.2):

1. Para que la cortina se ubique en esta posición la temperatura debe ser menor o igual a 12°C o, la velocidad del viento debe ser mayor a 15 Km/h. Cuando existe lluvia la cortina se ubica en esta posición, sin importar si ántes se cumple alguna de las condiciones para temperatura o viento.

2. La cortina debe estar en esta posición cuando la temperatura sea mayor a 12°C y menor o igual a 16°C o, la velocidad del viento sea mayor a 10Km/h y menor o igual a 15Km/h.

3. La cortina debe estar en esta posición cuando la temperatura sea mayor a 16°C y menor o igual a 18°C o, la velocidad del viento sea mayor a 5Km/h y menor o igual a 10Km/h.

4. Para esta posición la temperatura debe ser mayor a 18°C o, la velocidad del viento debe ser menor o igual a 5Km/h.

3.5.3. Despliegue de información en LCD

Para utilizar la pantalla de cristal líquido se hace uso del archivo *lcd.c*. Incluye funciones que son necesarias para la configuración de la pantalla y el envío de los datos para ser presentados en la LCD. Las funciones utilizadas se describen brevemente a continuación.

1. lcd_init(). Configura la LCD para usar 4 líneas del bus de datos, selecciona el tamaño de fuente, enciende y limpia la pantalla.

2. lcd_clear(). Limpia la pantalla y ubica el cursor en la primera posición para comenzar a escribir.

3. lcd_home(). Ubica el cursor en la primera posición.

4. lcd_putch(). Envía un caracter para ser mostrado en la pantalla. Si se envía como un caracter ASCII se debe indicar entre comillas simples (' ').

5. lcd_puts(). Envía una cadena de caracteres para ser mostrada en la pantalla. Si se envían como caracteres ASCII se deben indicar entre comillas dobles (" ").

6. lcd_goto(). Ubica el cursor en una posición específica para comenzar a escribir.

En la Figura 2.8 se encuentran las instrucciones básicas que se envían a la LCD mediante las funciones anteriores.

Los datos que recibe la LCD para ser mostrados son interpretados como caracteres ASCII, es decir, que el dato de 8 bits puede ser enviado en su equivalente binario, hexadecimal o decimal, pero lo más conveniente es enviarlo en formato ASCII.

Envío de temperatura y viento a la LCD

Para mostrar el valor de temperatura y velocidad de viento en la LCD se requiere de más de 1 dígito por ser cantidades decimales y mayores a 9. Para esto se utiliza un algoritmo que descompone el dato de tipo flotante en dígitos de 0 a 9 para las centenas, decenas, unidades, décimas, etc. Así se puede enviar cada dígito a la vez en formato ASCII y se forma la cantidad que se obtuvo de la medición con los dígitos necesarios.

La función *show_var* contiene este algoritmo y se explica enseguida.

- El dato tipo flotante *var* se divide por 100 y se almacena en la variable C como un dato de 0 a 9 para las centenas.
- El dato anterior C se multiplica por 100 para restar las centenas a *var*. Este nuevo resultado, que debe ser menor que 100, se divide por 10 y se almacena el resultado en D, que son las decenas.
- Se restan las centenas y las decenas a *var* y se tienen las unidades que se guardan en U.
- Se restan centenas, decenas y unidades a *var*, se multiplica por 10 y se obtienen las décimas que se guardan en *d*.

Cada dígito se envía de izquierda a derecha con la función *lcd_putch*. Los números en el formato ASCII comienzan en el valor `30h='0'`. Si se envía el valor del dígito a representar sin ser modificado ántes, entonces no se podrá visualizar en la LCD. Se debe sumar el valor 30h para obtener el equivalente ASCII para el dígito deseado. Por ejemplo: `lcd_putch('0'+X)` envía un caracter para representar un dígito de 0 a 9, donde `'0'=30h`; $0 \leq X \leq 9$.

El punto decimal también debe enviarse como caracter `('.')`, en la posición correcta, es decir, después de las unidades.

Las variables *posicion*, *sigpos* y *lluvia* son representadas en la LCD de la misma manera que se envía cada dígito con la función *lcd_putch*.

3.5.4. Activación del actuador

En esta sección del programa se acciona el dispositivo de potencia que enciende el motor acoplado a la cortina de ventilación. Para activar el motor en un sentido o en otro se utilizan sentencias *if* e *if-else* dentro de una estructura tipo *switch* que utiliza el valor de *sigpos* para evaluar cada caso.

Se designan los pines RC3 y RC4 como *cierra* y *abre* respectivamente (mediante un Puente H se puede controlar el sentido de giro de un motor con dos señales lógicas). Cada pin se activa dentro de una sentencia de lazo *do-while* y cuando la posición actual coincide con la posición deseada se desactiva el pin correspondiente, deteniendo el giro del motor.

Enseguida se describen los cuatro casos para la estructura switch.

1. Para el caso en que de acuerdo a las condiciones meteorológicas se determine que la siguiente posición debe ser 1 ($sigpos = 1 = 0\,\%abierto$), se compara el valor de la posición actual. Si el valor es mayor a 1, se cierra la cortina hasta que SW00=1, lo cual significa que la cortina ha llegado a la posición 1. Si la posición actual es 1, entonces no se realiza ninguna acción y sale de la estructura switch sin mover la cortina.

2. Para cuando la siguiente posición debe ser igual a 2 ($sigpos = 2 = 33\,\%abierto$), se consideran tres situaciones posibles según la posición actual:

   - posicion <2. En este caso lo que se debe hacer es abrir la cortina hasta que SW33=1.
   - posicion = 2. La cortina está en la posición correcta, por lo tanto no se realiza ninguna acción.
   - posicion >2. La cortina se debe cerrar hasta llegar a la posición 2 (SW33=1).

3. Este caso es parecido al anterior con tres situaciones según la posición actual cuando $sigpos = 3 = 66\,\%abierto$.

   - posicion <3. La cortina se debe abrir hasta que SW66=1.
   - posicion = 3. La cortina está en la posición correcta, por lo tanto no se realiza ninguna acción.
   - posicion >3. La cortina se debe cerrar hasta que SW66=1.

4. En este caso se tienen dos posibilidades: si la posición actual es igual a 4, sale de la estructura switch sin realizar ninguna acción ya que coincide con la posición siguiente; si la posición actual es menor se abre la cortina hasta que SW100=1

3.5.5. Envío de variables via serial

Para supervisar el estado del sistema se incluye una función que permite enviar los valores obtenidos de la conversión A/D y de las demás variables. Esta función está declarada como una interrupción, es decir, que para ser ejecutada no es necesario llamarla en alguna parte del programa, sino que se debe habilitar alguna de las interrupciones

con las que cuenta el microcontrolador y cuando ésta se activa el código se ejecuta.

Para utilizar el puerto de comunicación serial se usa el archivo *sci.c*, con el que se puede configurar el dispositivo y enviar o recibir datos.

Para configurar la comunicación via serial se llama la función sci_init(). Esta función es la encargada de encender el módulo de comunicación serial, seleccionar el tipo de transmisión como asíncrona, calcular la velocidad de la tasa de transmisión, seleccionar el modo de transmisión y recepción con 9 u 8 bits, habilitar la transmisión y recepción. El llamado de esta función requiere el paso de dos parámetros: la velocidad en baudios para la interfaz y una constante para indicar si se utilizan 9 u 8 bits en la comunicación.

Una vez configurada la interfaz es necesario configurar las interrupciones. Primero se habilita la interrupción que se desea utilizar (Sección 2.2.2), en este caso se habilita la interrupción por recepción via serial (RCIE del registro PIE1); se habilitan las interrupciones por periféricos y las interrupciones globales (PEIE y GIE del registro INTCON respectivamente). Por último es necesario limpiar la bandera de interrupción RCIF del registro PIR1. Es importante hacerlo en ese orden para evitar un disparo no deseado de alguna interrupción.

La función *send* es la encargada de enviar los datos de las variables involucradas en todo el proceso. El código se ejecuta al activarse la interrupción después de recibir algún dato en el puerto serial (pin RC7/RX/TD), el dato se almacena en el registro RCREG. El dato recibido se compara con una constante (puede ser el identificador del dispositivo o algún código de instrucción), si coincide se llama la función *sci_putbyte* para enviar una a una las variables *temp*, *vien*, *posicion*, *PORTC* y *PORTD*. Una vez enviadas las variables el programa sale de la rutina de interrupción.

Una consideración importante al entrar a alguna rutina de interrupción, es que se debe deshabilitar la interrupción específica o las interrupciones globales mientras se ejecuta la rutina y, se debe habilitar nuevamente en la última instrucción ántes de salir de la rutina. La bandera de interrupción que se ha activado se debe limpiar dentro de la rutina de interrupción preferentemente ántes de volver a habilitar la interrupción.

3.6. Código del programa para el PIC16F877

3.6.1. Programa principal

La primera parte de todo el código consiste en la declaración de las directivas del programa, la configuración del dispositivo para su grabación y la declaración de variables globales y funciones. En el programa principal se configuran los recursos internos

del PIC y se crea un ciclo continuo en el que se determina la posición hacia la que debe moverse la cortina (*sipos*) y se genera la señal de control para activar el actuador.

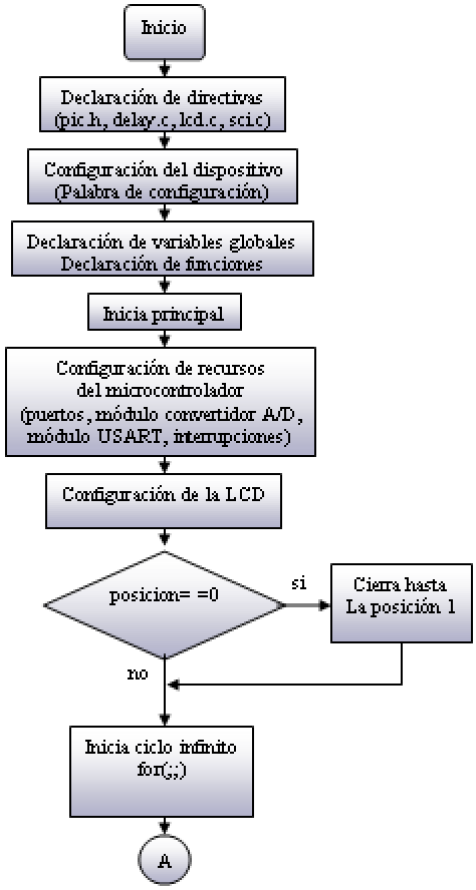

Figura 3.6: Diagrama de flujo del programa principal.

En la Figura 3.6, se muestra el diagrama de flujo para la primera parte del programa principal, donde se configura el dispositivo, los recursos internos y periféricos del microcontrolador.

En la Figura 3.7 se muestra el diagrama para el cálculo de la posición siguiente de acuerdo a las condiciones y los valores establecidos de temperatura exterior, velocidad de viento y de presencia de lluvia.

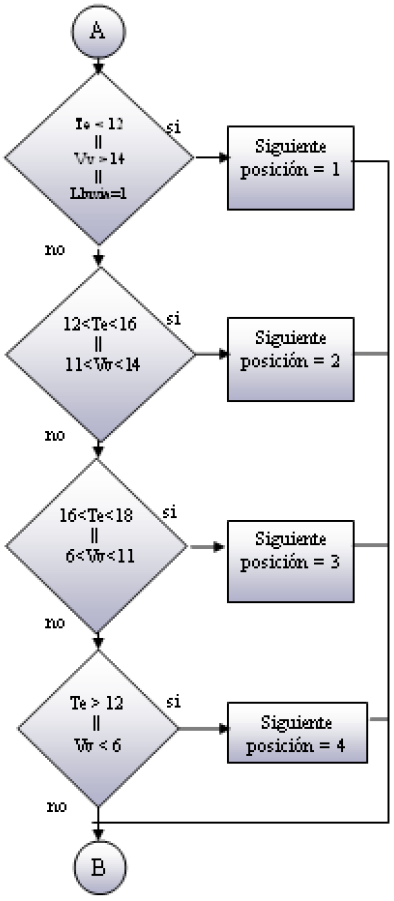

Figura 3.7: Cálculo de la posición siguiente.

Código del programa

Se incluye el código en C que se ha editado y compilado con MPLAB IDE [27] y PICCLITE [28] para grabarlo en el microcontrolador. Los comentarios se indican con `//comentario` o entre `/*comentario*/`.

```
/* Programa para control de cortinas de invernadero */

 #include   <pic.h>
 #include   "delay.c"
 #include   "lcd.c"
 #include   "sci.c"

__CONFIG(0x3D32);

 static bit cierra  @ (unsigned)&PORTC*8+3; // asigna RC3 a cierra
 static bit abre    @ (unsigned)&PORTC*8+4; // asigna RC4 a abre

 static bit SW100   @ (unsigned)&PORTD*8+7; // asigna RD7 a posicion4
```

```
 static bit SW66    @ (unsigned)&PORTD*8+6; // asigna RD6 a posicion3
 static bit SW33    @ (unsigned)&PORTD*8+5; // asigna RD5 a posicion2
 static bit SW00    @ (unsigned)&PORTD*8+4; // asigna RD4 a posicion1
 static bit lluvia  @ (unsigned)&PORTD*8+3; // asigna RD3 a lluvia
 static bit flag;

 void update(void);
 void show_var(float var);
 void delay_s(unsigned char sec);
 void interrupt send(void);

 unsigned char posicion,sigpos,temp,vien;
 float temperatura, viento;

void main(void)
 {
    TRISA =  0xff;
    TRISB = 0x00;
    TRISC = 0x80;
    TRISD = 0xff;

    ADCON1 = 0x00;
    ADCON0 = 0x80;
    DelayMs(1);
    ADON = 1;

    lcd_init();
    lcd_clear();
    lcd_home();

    flag = sci_init(9600,SCI_EIGHT);
    RCIE = 1;
    PEIE = 1;
    GIE  = 1;
    RCIF = 0;

    lcd_puts("READY");
    delay_s(5);
    lcd_clear();

    update();

if (posicion!=1){
    delay_s(10);                        // después de algún reset o
        do{                             // reinicio por corte de alimentación
            cierra = 1;                 // cierra la cortina sin importar
            update();                   // en que posición quedó
        }while(SW00 != 1);
        cierra = 0;
        abre   = 0;
        posicion = 1;
}

for(;;){
    update();
if((temperatura<=12)||(viento>15)){
    sigpos = 1;
    }else{
    if(((temperatura>12)&&(temperatura<=16))||((viento>10)&&(viento<=15))){
        sigpos = 2;
        }else{
        if(((temperatura>16)&&(temperatura<=18))||((viento>5)&&(viento<=10))){
            sigpos = 3;
            }else{
```

```
            if((temperatura>18)||(viento<=5)){
                sigpos = 4;
                }else{
                    sigpos = 0;
                }
            }
        }
    }

if(lluvia){                             // si hay lluvia no importa
    sigpos = 1;                         // cualquier otro valor de posición siguiente
}

delay_s(1);

switch(sigpos){

case (1):                               // abre cortina 0%
    if(posicion!=1){
        do{
            cierra = 1;
            update();
        }while(!SW00);
        cierra = 0;
        posicion = 1;
    }
    break;

case (2):                               // abre cortina 33%
    if(posicion<2){
        do{
            abre = 1;
            update();
        }while(!SW33);
        abre = 0;
        posicion = 2;
    }else{
        if(posicion>2){
            do{
                cierra = 1;
                update();
            }while(!SW33);
            cierra = 0;
            posicion = 2;
        }
    }
    break;

case (3):                               // abre cortina 66%
    if(posicion<3){
        do{
            abre = 1;
            update();
        }while(!SW66);
        abre = 0;
        posicion = 3;
    }else{
        if(posicion>3){
            do{
                cierra=1;
                update();
            }while(!SW66);
            cierra = 0;
            posicion = 3;
```

```
        }
    }
    break;

case (4):                               // abre cortina 100%
    if(posicion!=4){
        do{
            abre = 1;
            update();
        }while(!SW100);
        abre = 0;
        posicion = 4;
    }
    break;
 }                                      // switch cierra

 }                                      // ciclo for infinito cierra
 }                                      // main
cierra
```

3.6.2. Funciones

En la Sección 3.5 se describe de manera general cada una de las funciones utilizadas y que pueden ser llamadas desde el programa principal o desde alguna de las funciones. A continuación se muestra el código de cada función con algunos comentarios

Lectura y actualización de variables

La función *update* se encarga de actualizar las siguientes variables:

- Temperatura y viento mediante la conversión A/D.
- Posición y lluvia mediante la verificación de cada uno de los bits asignados a las variables.

Esta función también se encarga del despliegue de la información de todas las variables en la LCD después de que han sido actualizadas.

Código de la función

```
void update(void) {
    lcd_home();
    ADCON0 = (0 << 3) + 0x81;           // habilita ADC, RC osc, canal 0.
    DelayUs(255);
    ADGO = 1;
    while(ADGO)
        continue;                       // espera fin de conversión
    temp = ADRESH;
    temperatura = ((temp*4.9/256)-1.363636364)/0.072727273;

    ADCON0 = (1 << 3) + 0x81;           // habilita ADC, RC osc, canal 1.
    DelayUs(255);
    ADGO = 1;
    while(ADGO)
        continue;                       // espera fin de conversión
```

```
    vien = ADRESH;
    viento = ((vien*4.9/256)-1)/0.0444445;

if(SW100){                              // verifica el estado de las
    posicion=4;                         // cortinas en el bit
    }else{                              // correspondiente a 100%

    if(SW66){                           // verifica el estado de las
        posicion=3;                     // cortinas en el bit
        }else{                          // correspondiente a 66%

            if(SW33){                   // verifica el estado de las
                posicion=2;             // cortinas en el bit
                }else{                  // correspondiente a 33%

                    if(SW00){           // verifica el estado de las
                    posicion=1;         // cortinas en el bit
                }                       // correspondiente a 0%
            }
        }
    }

    lcd_puts("TE_");                    // envía el valor de cada
    show_var(temperatura);              // variable a la LCD
    lcd_goto(8);
    lcd_puts("VV_");
    show_var(viento);

    lcd_goto(40);
    lcd_puts("LL_");
    lcd_putch('0'+lluvia);
    lcd_puts(" P_");
    lcd_putch('0'+posicion);
    lcd_puts(" SP_");
    lcd_putch('0'+sigpos);
    lcd_putch(' ');

if(cierra){                             // pone un indicador en la
    lcd_puts("vv");                     // LCD de acuerdo a la orden que se
    }else{                              // envía para abrir o cerrar
    if(abre){                           // las cortinas
        lcd_puts("^^");
        }else{
            lcd_puts("--");
        }
    }
    DelayMs(1);
}
```

Envío de temperatura y viento a la LCD

Para poder enviar el valor decimal de estas dos variables se utiliza la función *show_var*, la cual descompone el valor en dígitos y los envía uno a uno para formar la cantidad obtenida de la medición en la LCD.

Código de la función

```
void show_var(float var)
 {
 unsigned char U, D, C, d;
```

```
    C = var/100;                        // obtiene centenas
    D = (var-(C*100))/10;               // obtiene decenas
    U = (var-(C*100)-(D*10));           // obtiene unidades
    d = (var-(C*100)-(D*10)-U)*10/1;    // obtiene décimas

    lcd_putch('0'+D);                   // envía cada dígito
    lcd_putch('0'+U);                   // de cada variable
    lcd_putch('.');
    lcd_putch('0'+d);

return;
 }
```

Envío de variables via serial

La función *send* se encarga de enviar las variables en formato de 8 bits mediante la interfaz serial una vez que se ha detectado una interrupción por recepción via serial y que se ha verificado que corresponde al identificador del dispositivo.

Código de la función

```
void interrupt send(void)
 {
    RCIE=0;
    if(RCREG==0x00){
        sci_putbyte(temp);
        sci_putbyte(vien);
        sci_putbyte(posicion);
        sci_putbyte(PORTC);
        sci_putbyte(PORTD);
    }
    RCIF=0;
    RCIE=1;
    return;
 }
```

3.7. Descripción del programa para LabVIEW

El programa utilizado en LabVIEW tiene como finalidad mantener un monitoreo constante mediante una computadora de las variables involucradas en el sistema, así como del estado de las cortinas de ventilación. El periodo de muestreo en LabVIEW es independiente del sistema mínimo y puede ser programado desde algunos segundos hasta varios minutos, según la cantidad de muestras que se desea obtener.

El programa puede dividirse en secciones para su descripción como sigue.

1. Configuración.
2. Envío y recepción de datos.
3. Conversión de variables.
4. Despliegue de información.

3.7.1. Configuración

En esta parte se configura el puerto serie de la computadora con la mismos parámetros del sistema mínimo (Sección 3.5.5). La mayoría de las computadoras cuentan con dos puertos (COM1 y COM2), por lo que se puede realizar la conexión en cualquiera de ellos siempre que se configure el puerto correcto.

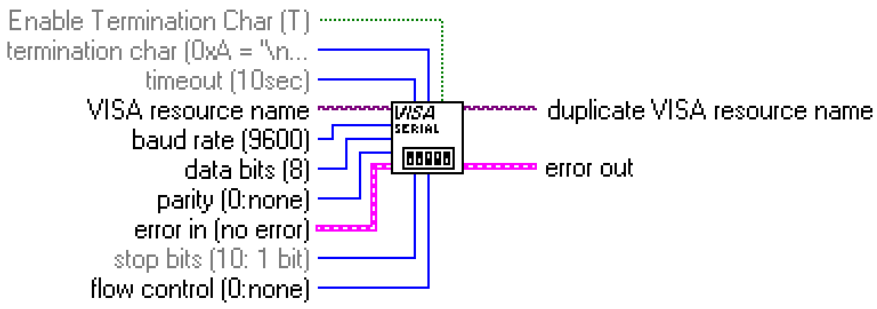

Figura 3.8: Función para configurar el puerto serie.

3.7.2. Envío y recepción de datos

Para actualizar los datos es necesario que se envíe un caracter al PIC, el cual puede ser interpretado como un ID, un código de instrucción, o ambos. Para esto se emplea la función *VISA write* (Figura 3.9), y se debe especificar el puerto serie y el dato a enviar en formato hexadecimal.

Figura 3.9: Función para escribir en el puerto serie.

Una vez que el PIC recibe y verifica que el identificador y/o la instrucción coinciden, envía los bytes que contienen los valores de las variables de temperatura exterior, velocidad de viento, lluvia, posición y estado del actuador (abre, cierra). Estos bytes son recibidos por la función *VISA read* (Figura 3.10), la cual requiere se especifique el puerto serie y la cantidad de bytes que debe recibir.

Figura 3.10: Función para leer desde el puerto serie.

3.7.3. Conversión de variables

Cada byte recibido debe ser modificado mediante funciones numéricas para que represente de manera correcta el valor de la variable a la cual corresponde. Los dos primeros bytes corresponden al valor obtenido de la conversión A/D de temperatura y viento por lo que debe obtenerse su valor en °C y en Km/h de acuerdo a las Ecuaciones 3.4 y 3.5.

El byte correspondiente a posición contiene el valor de la posición de las cortinas en el momento de que se envía, por lo que no es necesario modificar su valor ya que sólo puede tomar valores de 0 a 4. Los dos últimos bytes recibidos contienen el valor de cada uno de los bits del Puerto C y del Puerto D, aunque sólo se utilizan los que corresponden al estado del actuador (cierra, abre), y presencia de lluvia. Estos bits son interpretados como 0 ó 1 (cierto, falso).

3.7.4. Despliegue de información

Para visualizar el valor de las variables en el instante en que se toma una muestra se emplean indicadores numéricos y gráficos (instrumentos virtuales). Los instrumentos están distribuidos en un panel fontal (Figura 3.11), y puede ser modificada tanto su apariencia como su formato y precisión. Para temperatura se utiliza un indicador tipo termómetro con una escala de 0 a 50°C, para la velocidad del viento se utiliza un indicador con una escala de 0 a 50Km/h, para la posición de las cortinas se usa un indicador de barra con una escala de 1 a 4 y para lluvia, abre y cierra se emplean indicadores tipo led.

Figura 3.11: Indicadores en el Panel frontal en LabVIEW.

También se utilizan indicadores tipo histograma donde se puede observar el comportamiento de las variables durante un intervalo de muestras (Figura 3.12).

Cada una de las gráficas muestran los valores obtenidos de Temperatura exterior(a), Velocidad de viento (b), Presencia de lluvia (c), Abrir (d), Cerrar (e), Posición de las cortinas (f). Cada muestra se toma cada 2 segundos para formar cada histograma, aunque el periodo se puede modificar desde el programa en LabVIEW sin que se altere el funcionamiento del autómata. El programa completo en LabVIEW, puede encontrarse en el Apéndice B.

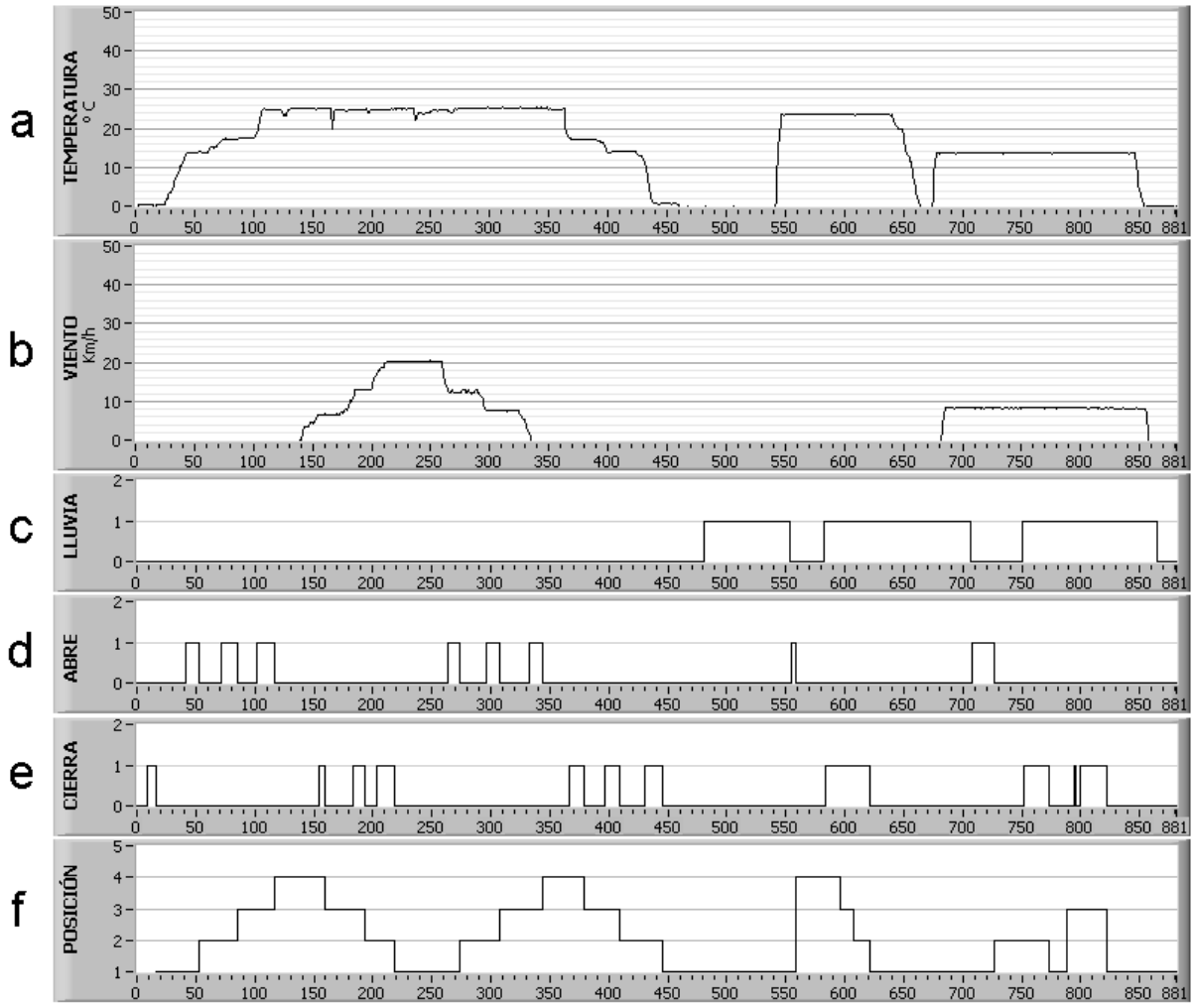

Figura 3.12: Histogramas en el Panel frontal en LabVIEW.

Capítulo 4

Pruebas de laboratorio y resultados

4.1. Análisis gráfico de las variables

En este capitulo se presentan las gráficas del comportamiento de las variables consideradas para diseñar el autómata y, que además se toman como referencia para simular el funcionamiento del control de apertura y cierre de las cortinas con MATLAB. También se muestran las gráficas obtenidas en LabVIEW de la simulación de las variables en tiempo real utilizando la interfaz serial RS-232.

4.1.1. Presencia de lluvia

La variable de presencia de lluvia se genera considerando los datos obtenidos de humedad y radiación solar, ya que la base de datos con que se cuenta no incluye esta variable.

En la Figura 4.1 se muestran las gráficas de humedad exterior, radiación solar y lluvia. De acuerdo a estas gráficas se asume que es un día soleado o con ausencia de lluvia cuando los valores de humedad son bajos y la radiación solar es alta. Por el contrario, cuando la humedad exterior es alta y la radiación solar es baja se considera que hay presencia de lluvia, y cuando la radiación solar es muy cercana a cero o cero, se considera que es de noche. Los valores que mostrados y que se consideran para obtener los datos de lluvia son aproximados debido a que se carece de alguna referencia, por lo que puede resulta necesario ajustarlos a las mediciones observadas durante un periodo de lluvia. Para la base de datos analizada, la gráfica indica la presencia de lluvia cuando adopta el valor de 1 y, ausencia de lluvia cuando es 0.

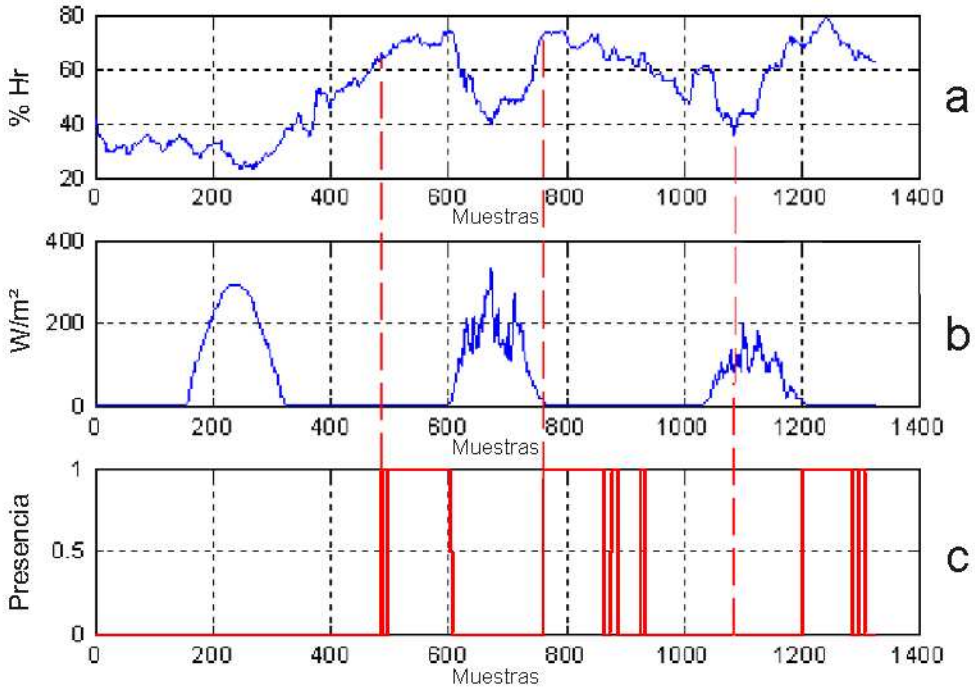

Figura 4.1: Presencia de lluvia: a)Humedad relativa, b)Radiación solar, c)Lluvia.

Condiciones que deben cumplirse para que *lluvia* sea igual a 1.

- Presencia de lluvia. Humedad exterior debe ser mayor o igual a 65 % y radiación solar debe ser menor o igual a 10.
- Ausencia de lluvia. Humedad exterior menor a 65 % y radiación solar mayor a 10.

4.1.2. Velocidad del viento

En la Figura 4.2 se observa el comportamiento de la Velocidad del viento de la base de datos del invernadero.

En la gráfica se puede observar que existen ráfagas que van desde 0 Km/h y repentinamente se elevan hasta los 14 Km/h, por lo que resulta necesario establecer rangos con histeresis, de lo contrario esto puede ocasionar que los cambios de posición de las cortinas sean muy repentinos, y que el motor no pueda activarse a tal velocidad, debido a que no es posible cambiar de giro con la misma velocidad con la que la ráfaga de viento pasa de un rango a otro.

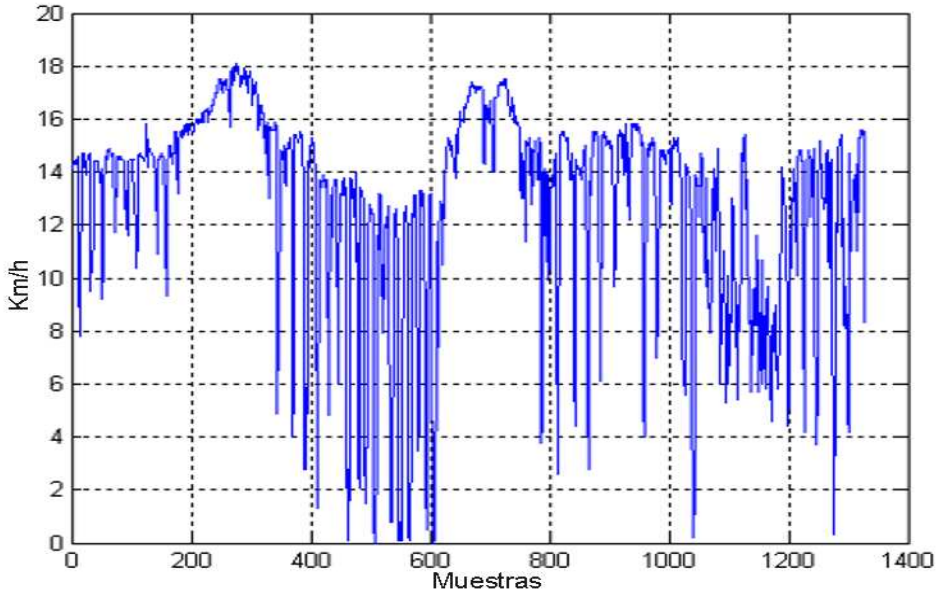

Figura 4.2: Gráfica de Velocidad del viento.

4.1.3. Temperatura

Como se mencionó en la Sección 3.2, los rangos de operación de temperatura considerados para el controlador de las cortinas, son obtenidos de la temperatura exterior para no ocasionar un conflicto con el controlador de calefacción. En la Figura 4.3 se puede observar el comportamiento del controlador de la calefacción, con respecto a los datos de temperatura interior. Como se muestra en ambas gráficas, cuando la temperatura interior es menor a 20°C la calefacción se activa y, cuando es superior a 24°C se desactiva.

En la Figura 4.4 se muestran la temperatura interior y la temperatura exterior. Aquí se puede observar la relación entre ellas al aumentar o disminuir la temperatura exterior y como la temperatura interior se mantiene por arriba de los 20°C por el efecto del calefactor instalado en el invernadero.

Tomando en cuenta la variación de temperatura exterior, se puede observar que existen valores menores a 12°C, para los cuales la cortina debe estar completamente cerrada con el fin de evitar perdidas de calor en el interior. Por el contrario, para valores mayores a 18°C en el exterior se pueden tener valores cercanos a 30°C en el interior, por lo que la cortina se debe mantener completamente abierta para permitir una buena ventilación.

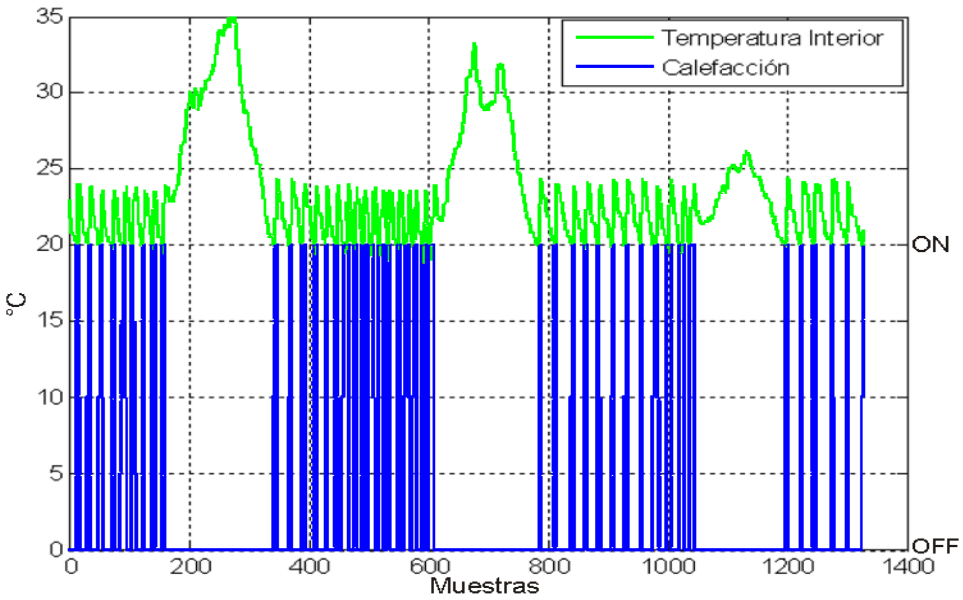

Figura 4.3: Calefacción y temperatura interior.

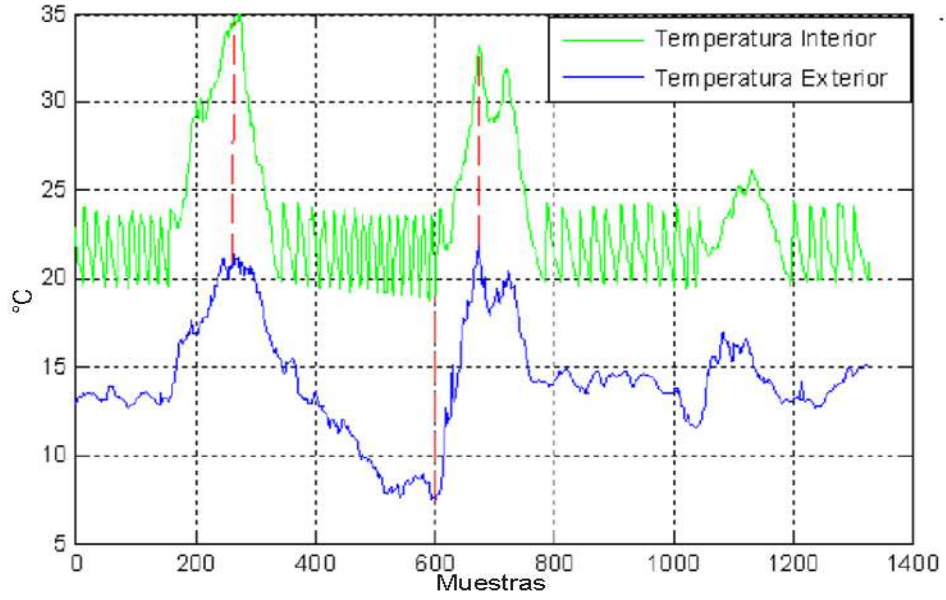

Figura 4.4: Temperatura interior y temperatura exterior.

4.1.4. Control de cortinas

La Figura 4.5 muestra el comportamiento de las cortinas respecto a las tres variables consideradas (presencia de lluvia, velocidad de viento y temperatura exterior). La escala vertical representa el porcentaje de apertura(0, 33 % abierto, 66 % abierto y 100 % abierto).

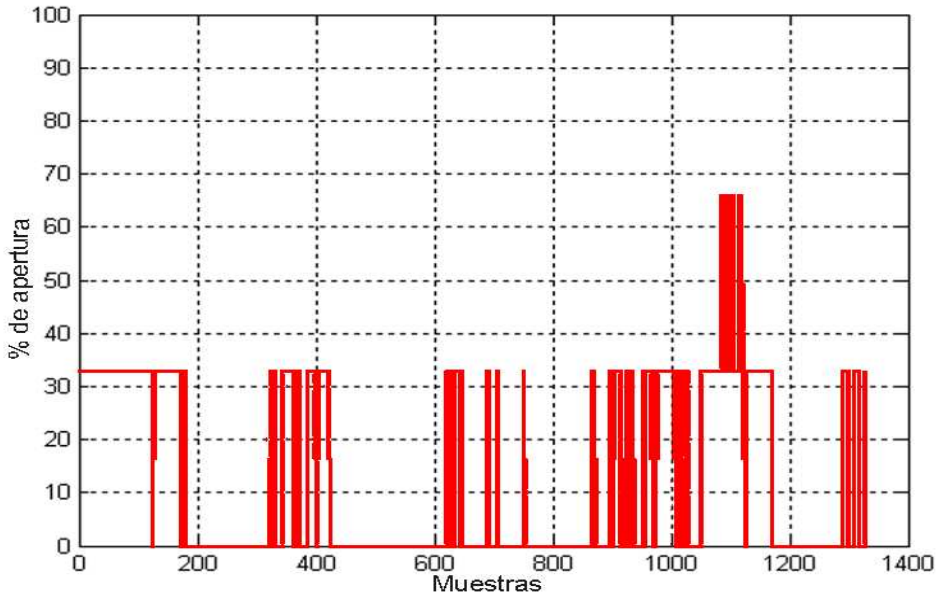

Figura 4.5: Comportamiento de las cortinas.

Se puede observar que para la base de datos utilizada, las cortinas no se llegan a abrir al 100 % al no cumplirse alguna de las condiciones necesarias de temperatura o viento. Por ejemplo, existen valores mayores a 18°C, para los cuales deberían abrirse 100 % las cortinas, pero al no ser menor o igual a 5Km/h la velocidad del viento, no llega a cumplirse dicha condición y no abre completamente la cortina para evitar daños al cultivo ocasionados por la velocidad del viento.

La figura 4.6 muestra la gráfica donde se pueden observar las tres variables (temperatura exterior, velocidad de viento y presencia de lluvia), así como el comportamiento del controlador de las cortinas respecto a ellas.

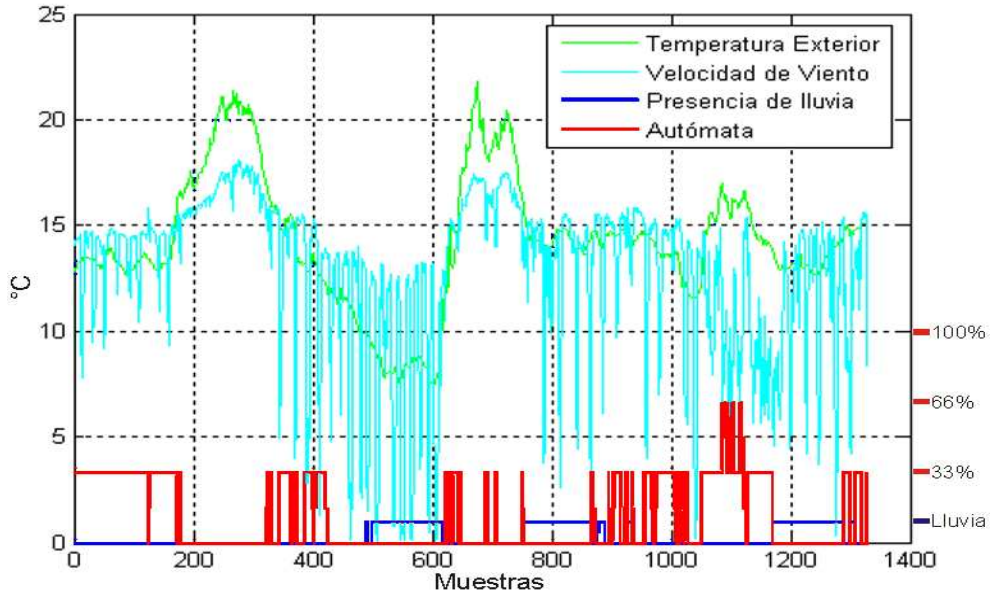

Figura 4.6: Comportamiento del autómata respecto a las variables involucradas.

4.2. Prueba del autómata

Para realizar las pruebas de funcionamiento del dispositivo, es necesario recrear una serie de condiciones que cubran todos los casos posibles. Para esto se dispone de una tablilla de pruebas en la que se conectan dos circuitos divisores de tensión con potenciómetros (para simular variaciones de *Temperatura* y *Velocidad de viento* en todo su rango), interruptores normalmente abiertos (para obtener las entradas digitales correspondientes a *Lluvia* y *Posición de las cortinas*) y diodos led (para indicar cuando *abre* o *cierra* la cortina).

En la Figura 4.7, se muestra el diagrama esquemático del circuito que se utiliza para la prueba del dispositivo y el nombre de cada terminal a la que se conecta en el sistema mínimo.

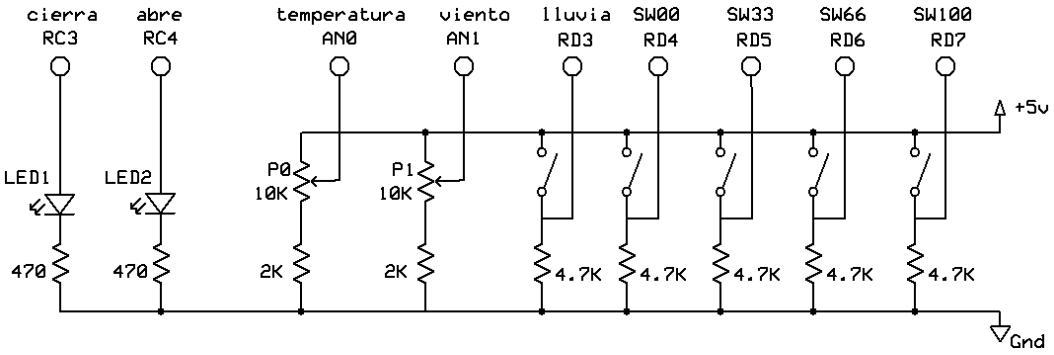

Figura 4.7: Circuito de prueba.

4.2.1. Condiciones de temperatura y velocidad de viento

Antes de comenzar a verificar el funcionamiento del autómata es necesario revisar el estado de las conexiones de cada uno de los periféricos y circuitos hacia la placa del sistema mínimo, así como verificar que el voltaje de la fuente de alimentación sea de 5V. De no ser así es necesario sustituir el regulador de voltaje o modificar el valor de referencia para la conversión A/D en el programa del PIC para disminuir el error de cuantificación durante la conversión A/D.

A continuación se describe una metodología general para verificar el funcionamiento del circuito para el control automático de las cortinas de acuerdo a las variables de temperatura y velocidad de viento. Es importante recordar que, a mayor temperatura el porcentaje de apertura es mayor y, por el contrario a mayor velocidad de viento el porcentaje de apertura es menor.

1. Colocar los potenciometros en la posición más baja y dejar los interruptores abiertos o en 0.

2. Alimentar el dispositivo y esperar que aparezcan en la pantalla LCD los indicadores de cada una de las variables en 0. Después de pocos segundos `SP_0` debe cambiar por `SP_1` y en la esquina inferior derecha aparece `'\/\/'` para indicar el estado del actuador, lo cual quiere decir que el pin cierra=1 y la cortina esta cerrando.

3. Colocar a 1 el interruptor `SW00` y esperar a que el indicador de posición y del actuador se actualicen en la LCD (`P_1` y `'--'`). Esto indica que la cortina llegó a la posición esperada o estado inicial y el actuador se desactiva con cierra=0.

4. Girar el potenciómetro de Temperatura hasta que `TE_` muestre un valor mayor a 12 y menor o igual a 16. `SP_` cambia a la posición siguiente y aparece `'/\/\'` al

activar el actuador de la cortina con abre=1. SW00 (posición 1) se debe colocar a 0 y SW33 (posición2) a 1 para que P_ y SP_ tengan el mismo valor.

5. Repetir el paso anterior para cada rango de temperatura (16<TE≤18, TE>18) activando el interruptor que se indique en SP_ hasta que el indicador '∧∧' o '∨∨' desaparezca.

6. Colocar el potenciómetro de Temperatura en un valor mayor a 18 para llegar a la Posición 4 y girar el potenciómetro de Velocidad de viento hasta que VV_ muestre un valor mayor a 5 y menor o igual a 10. SP_ disminuye y el indicador '∨∨' aparece, entonces se pone a 1 SW66 (posición3) hasta que aparezca P_3 y '∨∨' desaparezca.

7. Se continua girando el potenciómetro de Velocidad de viento para cerrar la cortina de acuerdo a los demás rangos (posición 2 para 10<VV≤15 y posición 1 para VV>15).

El procedimiento anterior se lleva a cabo modificando una sola variable por cada uno de los rangos a la vez, pero se puede realizar sin problema cambiando el valor de una variable hasta la posición deseada y modificando la otra variable para que vaya a una posición mayor o menor que la actual.

4.2.2. Condiciones de lluvia

Para probar las condiciones de presencia de lluvia se debe tener en cuenta que las cortinas deben cerrarse completamente al detectarse presencia de lluvia sin importar alguna otra condición de temperatura o velocidad de viento.

El siguiente procedimiento puede llevarse a cabo a partir del estado inicial (realizar el procedimiento anterior hasta el paso 3) o desde cualquier otro estado.

1. Durante el estado inicial (LL_0, P_1, SP_1) se pone a 1 el interruptor de lluvia y solo se debe modificar el indicador de lluvia con LL_1 ya que la cortina esta completamente cerrada.

2. Durante cualquier estado donde la posición sea diferente de 1 se activa el interruptor de lluvia. Los indicadores se modifican a LL_1, SP_1 y el indicador '∨∨' aparece, lo que indica que la cortina se debe cerrar hasta que llegue a la posición 1 (P_1) y '∨∨' desaparece.

Otro caso que difícilmente puede presentarse durante el funcionamiento normal, pero que puede servir para comprobar el cierre de las cortinas al existir presencia lluvia

es, cuando estando en la posición 1 (`P_1`) y el indicador de lluvia activado (`LL_1`), se activa algún otro interruptor de posición (`SW33`, `SW66`, `SW100`). Cuando esto sucede se activa el actuador para cerrar las cortinas y el indicador `'\/\/'` aparece hasta que se han cerrado completamente.

4.3. Prueba del autómata con LabVIEW

El automata puede funcionar independientemente de LabVIEW ya que cuenta con la pantalla LCD, pero aún así es conveniente contar con los datos de las variables involucradas en todo el proceso ya sea para su supervisión o, para un análisis más detallado.

Debido a que LabVIEW se utiliza como una herramienta de visualización del estado del autómata no se describe una metodología para el manejo del programa durante el funcionamiento del sistema, solo se mencionan los pasos y las consideraciones necesarias para que el programa se ejecute sin problemas.

- Verificar que la configuración del puerto en el programa sea la misma con la que se programó el microcontrolador o, de lo contrario la comunicación no podrá llevarse a cabo.
- Es recomendable conectar el convertidor RS-232 entre el puerto SCI del sistema mínimo y COM1 ó COM2 de la computadora antes de encender el dispositivo.
- Una vez encendido el dispositivo, se da la instrucción de ejecutar el programa y los datos de las variables se actualizan cada vez que se cumple el periodo de muestreo, sin importar la acción que se encuentre realizando el microcontrolador.

Una vez que el programa se esta ejecutando se puede comprobar cada condición siguiendo el mismo procedimiento que se describió en la sección anterior para las variables de Temperatura, Velocidad de viento y Lluvia. Cada vez que se modifique alguna variable en el dispositivo se debe esperar a que se actualice el valor en la pantalla del Panel Frontal de LabVIEW.

La Figura 4.8 muestra las gráficas obtenidas durante el procedimiento de prueba. Las muestras se tomaron cada 2 segundos. Las gráficas se describen por intervalos de muestras (líneas en color rojo), los cuales coinciden con los pasos del procedimiento de prueba descrito anteriormente.

00-20: Corresponde al encendido del dispositivo hasta llegar al estado inicial. Se puede observar que la única acción es cerrar cortina completamente (posición 1).

21-70: En este intervalo aumenta la temperatura hasta un valor entre 12°C y 16°C . La cortina debe abrir hasta el 33 % (posición 2).

71-140: La temperatura va aumentando hasta cubrir los rangos correspondientes a 66 % y 100 % de apertura. Las cortinas abren hasta una posición y se detienen.

141-260: Con un valor de temperatura mayor a 18°C , se aumenta el valor de la velocidad de viento para llegar a cada rango y permitir que la cortina cierre a cada una de las posiciones hasta cerrar completamente.

261-360: La velocidad de viento se disminuye hasta 0 y las cortinas deben de abrir 100 %.

361-480: La temperatura disminuye hasta cero para cerrar la cortina completamente.

481-550: Se activa el indicador de presencia de lluvia y aunque la temperatura o la velocidad del viento muestran un valor para el cual las cortinas deban estar abiertas, la grafica de abrir se mantienen en 0, dejando la cortina completamente cerrada.

551-580: En el momento en que el indicador de lluvia es 0 la cortina abre hasta llegar a la posición que corresponda al valor de las variables en ese momento.

581-780: Para cualquier posición mayor a 1, si hay lluvia, la cortina se cierra completamente aunque la temperatura y velocidad de viento se modifiquen.

781-880: Una vez que las cortinas se han cerrado completamente por presencia de lluvia, se observa como se activa una posición mayor y el indicador para cierre se vuelve a activar hasta llegar a la posición 1.

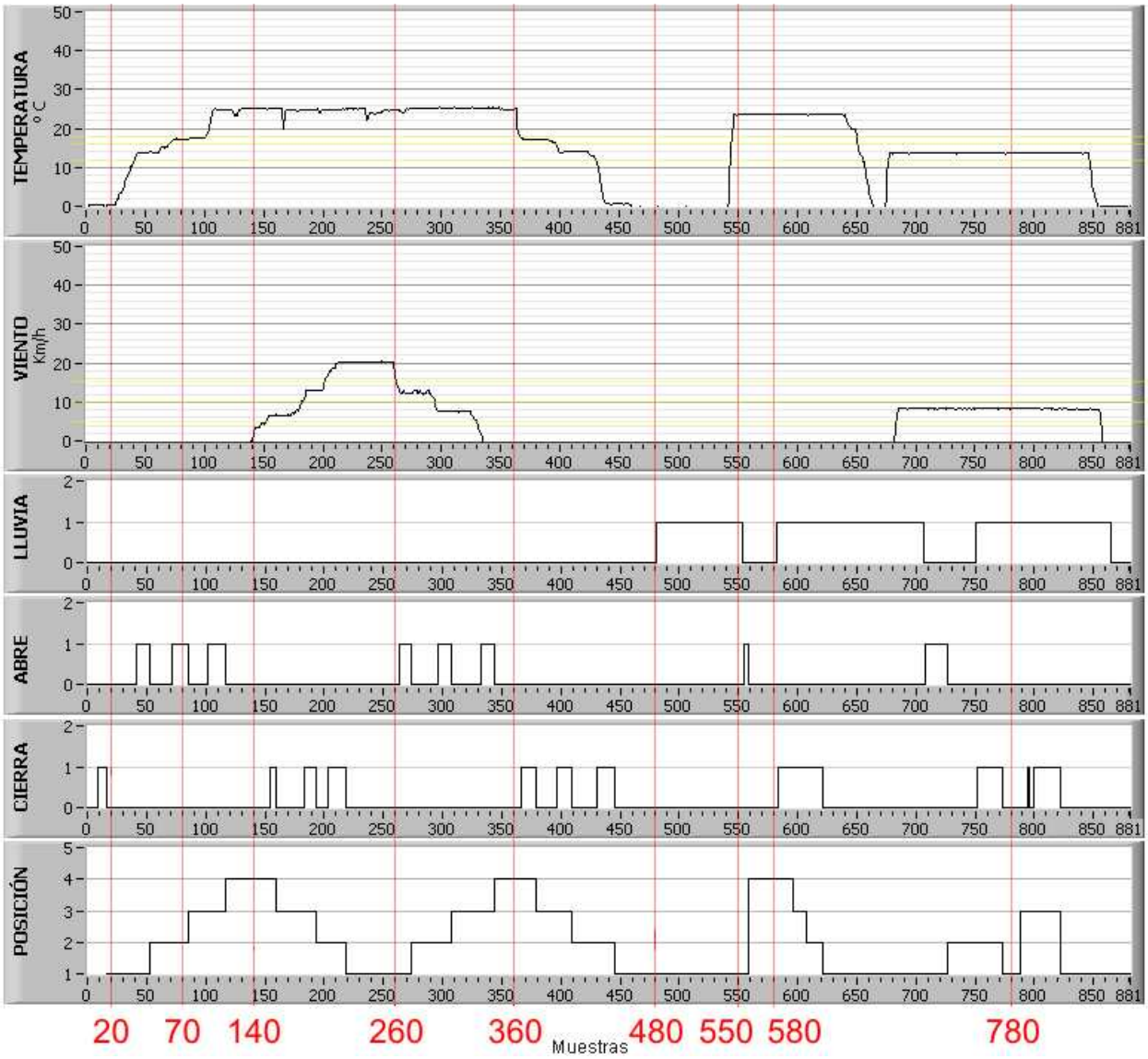

Figura 4.8: Comportamiento del autómata en LabVIEW.

Conclusiones y perspectivas

Conclusiones

El algoritmo creado evalúa las señales simuladas de temperatura exterior, velocidad de viento y presencia de lluvia y genera las señales de salida que corresponden a la acción que debe realizar el sistema de apertura y cierre de cortinas de un invernadero.

Durante el diseño, creación, pruebas y puesta en marcha del dispositivo se obtuvieron las siguientes conclusiones.

- El crear el programa para el microcontrolador con lenguaje C, permite adaptar el AFD fácilmente gracias al manejo de datos de tipo *float* (con punto decimal) y, al empleo de operaciones aritméticas con datos de este tipo.
- Los rangos de operación propuestos para cada variable funcionan correctamente en la simulación con MATLAB para la base de datos utilizada y para la simulación analógica en tiempo real con el sistema mínimo.
- Las variables y los estados se pueden visualizar en la LCD con la que cuenta el sistema mínimo.
- El uso de LabVIEW como herramienta de monitoreo facilita la visualización de los parámetros y variables mediante instrumentos virtuales.

Perspectivas

Algunos puntos que son considerados para mejorar el autómata a futuro son:

- La implementación del sistema autómata en un invernadero real, para realizar pruebas y mediciones en tiempo real.
- Debido a que el sistema mínimo cuenta con 5 entradas analógicas, y sólo se utilizaron dos de ellas (para temperatura y velocidad de viento), es posible utilizar las entradas que quedan disponibles para convertir las señales provenientes de sensores de humedad relativa exterior, radiación solar y temperatura interior.
- La posibilidad de modificar los rangos de referencia para abrir y cerrar las cortinas en el microcontrolador desde LabVIEW, ya que esto permitirá adaptarlo a los requerimientos del tipo de cultivo, a los niveles de crecimiento y a las estaciones del año.
- La creación de una base de datos desde LabVIEW para su análisis posterior con ayuda de herramientas de análisis estadístico.

Referencias

[1] *Tpagro.* http://www.tpagro.com, 2005.

[2] *Diseño de Invernaderos.* http://agrarias.tripod.com/invernaderos.htm, 2004.

[3] Camilo Montoya Parga. *Invernaderos o Macrotúneles.* http://www.aguascalientes.gob.mx/codagea/modulos/sitio/apoyos, 2005.

[4] Eduardo A. Sciutto; Juan E. Mascareña. *Sistema de Control Automático de Temperatura y Ventilación de un Invernadero.* Universidad Nacional de la Patagonia, San Juan Bosco, http://www.invernadero.8m.net/index.html, 2000.

[5] H. Challa; O. Körner. *Process-based Humidity Control Regime for Greenhouse Crops.* ELSEVIER, 2005.

[6] Antonio Matallana Gonzalez; Juan Ignacio Montero Camacho. *Invernaderos: Diseño, Construcción y Ambientación.* Mundi Prensa, Madrid, 1993.

[7] *Inge.* http://www.inge.cogia.net/index.php, 2005.

[8] Michael Kassler. *Agricultural Automation in the New Millennium.* Computers and Electronics in Agriculture, ELSEVIER, 2005.

[9] Nick Sigrimis; Robert E. King. *Advances in Greenhouse Environment Control.* ELSEVIER, 2005.

[10] G. van Straten; H. Challa; F. Buwalda. *Towards User Accepted Optimal control of Greenhouse Climate.* ELSEVIER, 2005.

[11] M.J. Lees;P.C. Young;W. Tych;A. Chotai;Z.S. Chalabi. *Design and Implementation of a Proportional-Integral-Plus (PIP) Control system for Temperature, Humidity and Carbon Dioxide in a Glasshouse.* Acta Horticulturae, 1996.

[12] Th. H. Gieling; W. Th. M. van Meurs; H. J. J. Janssen. *A Computer Nerwork with SCADA and a Case Tools for On-line Process Control in Greenhouses.* Pergamon, 1996.

[13] Kastuhiko Ogata. *Ingeniería de Control Moderna.* Prentice Hall, 1980.

[14] Benjamin C. Kuo. *Sistemas de Control Automático.* CECSA, 1989.

[15] Carlos Mario Velez. *Control Automático.* Escuela de Administración y Finanzas y Tecnología (EAFIT), Medellín, Colombia, http://www.control-systems.net/grupo/cmvelez.htm, 2004.

[16] Howard L. Harrison. *Controles Automáticos.* Trillas, 1978.

[17] Albert D. Helfrick; Wiliam D. Cooper. *Instrumentación Electrónica Moderna y Técnicas de Medición.* Prentice-Hall, 1990.

[18] W. Stanley ;F. Richard;Smith M. *Guía para Mediciones Electrónicas y Prácticas de Laboratorio.* 1995.

[19] *Material Auxiliar de Clase de Dispositivos Electrónicos.* http://www.el.uma.es/Disp_Electr/, 2006.

[20] Ronald J. Tocci. *Digital systems. Principles and Aplications.* Prentice-Hall, 1991.

[21] James W. Gault;Russell L. Pimmel. *Sistemas Digitales Basados en Microcontrolador.* Prentice-Hall, 1995.

[22] Ignacio Moreno Velasco. *Instrumentación Electrónica.* Area de Tecnología Electrónica. Universidad de Burgos, 2004.

[23] *Wikipedia.* http://es.wikipedia.org/wiki/, 2005.

[24] The MathWorks. *MATLAB User's Guide.* The MathWorks, Massachusetts, 1995.

[25] Felipe Gómez C. *Mastering MATLAB. A Comprehensive Tutorial and Reference.* Prentice-Hall, 1996.

[26] A. Sjoberg;B. Melin;P. Isaksson. *The MATLAB Handbook.* Addison-Wesley, New York, 1996.

[27] *MPLAB IDE Manual.* MICROCHIP, http://www.microchip.com/, 2005.

[28] *PICC Lite C Manual.* HI-TECH Software, http://www.htsoft.com/, 2004.

[29] Román Sosa;Francisco De Sande. *Tutorial Online para Teoría de Autómatas y Lenguajes Formales.* Universidad de la laguna, 2005.

[30] *LabVIEW User Manual.* National Instruments, http://ni.com/manuals, 2005.

[31] José Luis Rayon. *Termómetro digital.* http://www.monografias.com, 2005.

[32] José Ma. Angulo Usategui; Ignacio Angulo Martínez. *Microcontroladores PIC. Diseño Práctico de Aplicaciones.* McGraw-Hill, 2005.

[33] *PIC16F87X Data Sheet.* MICROCHIP, http://www.microchip.com/, 2005.

[34] *Aplicación Industrial de Micros.* http://www.geocities.com/micros_uan/index.html, 2005.

[35] Craig Peacock. *The Extended Concise LCD Data Sheet.* http://www.beyondlogic.org, 1999.

[36] Franz E. Soto Ramirez. *Diseño y Construcción de una Unidad Meteorológica basada en el Microcontrolador PIC16F877, Tesis de Ingeniería.* Universidad Autónoma del Estado de Hidalgo., 2006.

[37] Ma. de Lourdes Villanueva Vega. *Diseño de un Autómata para el Control de Variables Ambientales en Invernaderos, Reporte de Maestría.* Universidad Autónoma del Estado de Hidalgo, 2005.

[38] Ramón Brena. *Autómatas y Lenguajes. Un Enfoque de Diseño.* Tecnológico de Monterrey, 2003.

[39] D. Kelley. *Teoría de Autómatas y Lenguajes Formales.* Prentice-Hall Hispanoamericana, 1995.

Glosario

ASCII. (American Standard Code for Information Interchange o Código Estadounidense Estándar para el Intercambio de Información). Código de caracteres basado en el alfabeto latino tal como se usa en inglés moderno y en otras lenguas occidentales. Utiliza 7 bits para representar los caracteres, aunque inicialmente empleaba un bit adicional (bit de paridad) que se usaba para detectar errores en la transmisión.

Autómata Dispositivo o conjunto de reglas que realizan un encadenamiento automático y continuo de operaciones capaces de procesar una información de entrada para producir otra de salida.

Baudios. Unidad informática que se utiliza para cuantificar el número de cambios de estado, o eventos de señalización, que se producen cada segundo durante la transferencia de datos. La velocidad de transferencia de datos puede medirse en baudios o en bit/segundo. Lo habitual, hoy por hoy, es medirla en bits por segundo.

Bluetooth. Norma que define un estándar global de comunicación inalámbrica que posibilita la transmisión de voz y datos entre diferentes equipos mediante un enlace por radiofrecuencia.

Calibración. Procedimiento de comparación entre lo que indica un instrumento y lo que *debiera indicar* de acuerdo a un patrón de referencia con valor conocido.

Capacitancia. Capacidad o propiedad de un conductor de adquirir carga eléctrica cuando es sometido a un potencial eléctrico con respecto a otro en estado neutro. La capacitancia queda definida numéricamente por la carga que adquiere por cada unidad de potencial.

CAS. Circuito acondicionador de señal.

Cenital. Perteneciente o relativo al cenit, que es la intersección de la vertical de un lugar con la esfera celeste, por encima de la cabeza del observador.

Codificador. Circuito combinacional con 2^n entradas y n salidas cuya misión es presentar en la salida el código binario correspondiente a la entrada activada.

Compilador. Acepta programas escritos en un lenguaje de alto nivel y los traduce a otro lenguaje, generando un programa equivalente independiente, que puede ejecutarse tantas veces como se quiera.

Contador. Circuito secuencial construido a partir de biestables y puertas lógicas capaz de realizar el cómputo de los impulsos que recibe en la entrada destinada a tal efecto, almacenar datos o actuar como divisor de frecuencia. El cómputo se realiza en un código binario.

Convertidor analógico/digital. Convierte el voltaje analógico a su forma digital equivalente. La salida del convertidor A/D se puede desplegar visualmente y estar disponible como voltaje en pasos discretos para procesamiento posterior o grabación en un registrador digital.

Criptogámicas. Dicho de un vegetal o de una planta: Que carece de flores.

DCE. (Data Communication Equipment) Equipo de comunicación de datos.

Decodificador. Circuito combinacional, que convierte un código de entrada binario de N bits en M líneas de salida (N puede ser cualquier entero y M es un entero menor o igual a 2^N), tales que cada línea de salida será activada para una sola de las combinaciones posibles de entrada.

DLL. Dynamic Linking Library (Bibliotecas de Enlace Dinámico), término con el que se refiere a los archivos con código ejecutable que se cargan bajo demanda del programa por parte del sistema operativo. Esta denominación se refiere a los sistemas operativos Windows siendo la extensión con la que se identifican los ficheros, aunque el concepto existe en prácticamente todos los sistemas operativos modernos.

DTE. (Data Terminal Equipment) Equipo terminal de datos.

E/S. Entrada / salida.

EEPROM. (Electrically-erasable programmable read-only memory o Memoria de solo lectura programable y borrable eléctricamente) Estos dispositivos suelen comunicarse mediante protocolos como I^2C, SPI y Microwire. En otras ocasiones se integra dentro de chips como microcontroladores y DSPs para lograr una mayor rapidez. La memoria flash es una forma avanzada de EEPROM.

Esporas. Espora en biología designa un mecanismo reproductivo, generalmente haploide y unicelular. La reproducción por esporas permite al mismo tiempo la dispersión y la supervivencia por largo tiempo en condiciones adversas.

Feedback. Realimentación.

Fitosanitarios. Producto químico u orgánico para hacer frente a plagas, caracoles y limacos y todo tipo de enfermedades de las plantas, de efecto preventivo y curativo.

Fotosíntesis. Procesos metabólicos de los que se valen las células para obtener energía.

Humedad Relativa. Es la relación entre el porcentaje de la relación parcial del vapor de H_2O en la atmósfera real sobre el porcentaje de la presión del vapor H_2O en la temperatura ambiente. Representa la capacidad del aire de humedecer o secar materiales.

Inductancia. Relación entre la cantidad de flujo magnético, que lo atraviesa y la corriente I, que circula por ella.

IrDA. es un estándar que define una forma de implementar el uso e la tecnología infrarroja. Esta tecnología, basada en rayos luminosos que se mueven en el espectro infrarrojo. Los estándares IrDA soportan una amplia gama de dispositivos eléctricos, informáticos y de comunicaciones, permite la comunicación bidireccional entre dos extremos a velocidades que oscilan entre los 9.600 bps y los 4 Mbps.

LCD. (Liquid Crystal Display o Pantalla de Cristal Líquido)

Se trata de un sistema eléctrico de presentación de datos formado por 2 capas conductoras transparentes y en medio un material especial cristalino (cristal líquido) que tienen la capacidad de orientar la luz a su paso.

Cuando la corriente circula entre los electrodos transparentes con la forma a representar (por ejemplo, un segmento de un número) el material cristalino se reorienta alterando su transparencia.

LINUX. Nombre de un núcleo, uno de los componentes de un sistema operativo, que pretende ser una implementación libre de UNIX. Software libre (y de código abierto), donde el código fuente está disponible públicamente y cualquier persona, con los conocimientos informáticos adecuados, puede libremente estudiarlo, usarlo, modificarlo y redistribuirlo.

MAC. Nombre de una serie de ordenadores fabricados y comercializados por Apple Computer desde 1984.

Multiplexión. Técnica para transmitir varias señales secuencialmente a través un canal único.

PIC. (Controlador de interfaz de acoplamiento de periféricos) Microcontrolador o familia de microcontroladores tipo RISC.

Prototipo. Se puede referir a cualquier tipo de máquina en pruebas, o un objeto diseñado para una demostración de cualquier tipo.

PWM. (modulación por anchura de pulsos) Técnica utilizada para regular la velocidad de giro de los motores eléctricos. Mantiene el par motor constante y no supone un desprovechamiento de la energía eléctrica. Se utiliza tanto en corriente continua como en alterna, como su nombre lo indica, el establecer un tiempo en alto y un tiempo en bajo controlado, normalmente por relevadores (baja frecuencia) o MOSFET (alta frecuencia).

RAM. Random Access Memory (memoria de acceso aleatorio).

Se trata de una memoria de semiconductor en la que se puede tanto leer como escribir información. Es una memoria volátil, es decir, pierde su contenido al desconectar la energía eléctrica. Se utiliza normalmente como memoria temporal para almacenar resultados intermedios y datos similares no permanentes.

Radiación solar. Se conoce por radiación solar al conjunto de radiaciones electromagnéticas emitidas por el Sol que alcanzan la superficie de la Tierra. Éstas van desde el infrarrojo hasta el ultravioleta.

Reed-switch. Sensores magnéticos de tipo mecánico. Tienen gran difusión al emplearse en muy bajos voltajes, con lo que sirven de indicador de posición a PLCs y de indicador de posición de los cilindros neumáticos de émbolo magnético.

Redes neuronales. Modelos analógicos que tienen como objetivo reproducir en la medida de lo posible las características y la capacidad de procesamiento de información del conjunto de neuronas presentes en el cerebro de los seres vivos.

Registro. Es una memoria de alta velocidad que almacena valores usados en operaciones matemáticas.

RISC. Reduced Instruction Set Computer (Computadora con Conjunto de Instrucciones Reducido). Tipo de microprocesadores con las siguientes características fundamentales: 1.- Instrucciones de tamaño fijo y presentadas en un reducido número de formatos. 2.-Sólo las instrucciones de carga y almacenamiento acceden a la memoria por datos.

RST. Reset

Sensor. Dispositivo encargado de la toma de lectura de los parámetros ambientales, convirtiendolos en un voltaje analógico. Estos se clasifican básicamente en base al principio de funcionamiento.

Sistemas de adquisición. Son utilizados para medir y registrar señales eléctricas de manera directa o a través de transductores.

SPI. (Serial Peripheral Interface o Interfaz de comunicación serial entre periféricos) Incluye una línea de reloj, dato entrante, dato saliente y un pin de chip select, que conecta o desconecta la operación del dispositivo con el que uno desea comunicarse. De esta forma, este estándar permite multiplexar las líneas de reloj y datos de forma que se consigue un ahorro de pines.

Stack. Es una zona de la memoria en donde se guardan cosas. Por ejemplo las variables locales de las subrutinas y funciones, los parámetros, algunos resultados intermedios de cálculos complejos, etc.

Todo el manejo del stack lo realiza el compilador de forma automática, así que no hace falta preocuparse salvo cuando se acaba el lugar y el programa genera un error de desborde de pila.

TCP/IP. Se compone del Protocolo de Control de Transmisión (TCP) y el Protocolo de Internet (IP). El TCP/IP es la base de Internet, y sirve para enlazar computadoras que utilizan diferentes sistemas operativos, incluyendo PC, minicomputadoras y computadoras centrales sobre redes.

Termopar. Circuito formado por dos metales distintos que produce un voltaje siempre y cuando los metales se encuentren a temperaturas diferentes, los termopares son ampliamente usados como sensores de temperatura.

Transductores. Dispositivo capaz de transformar o convertir un determinado tipo de energía de entrada, en otra diferente de salida. Transforma parámetros físicos en señales eléctricas aceptables par el sistema de adquisición.

UDP. Protocolo del nivel de transporte basado en el intercambio de datagramas. Permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera.

UNIX. Sistema operativo portable, multitarea y multiusuario. Familia de sistemas operativos que comparten unos criterios de diseño e interoperabilidad en común.

USART. (Universal Synchronous Asynchronous Receiver Transmitter) El módulo receptor transmisor síncrono asíncrono universal es uno de los dos módulos de entrada/salida serial, tambien se conoce como interfaz de comunicación serial.

USB. (Bus de Serie Universal) Interfaz que provee un estándar de bus serie para conectar dispositivos a un ordenador personal (generalmente a un PC).

VIs. (Virtual Instruments) Programa en LabVIEW que modela la apariencia y el funcionamiento de un instrumento físico.

Apéndice A

Fotos del autómata

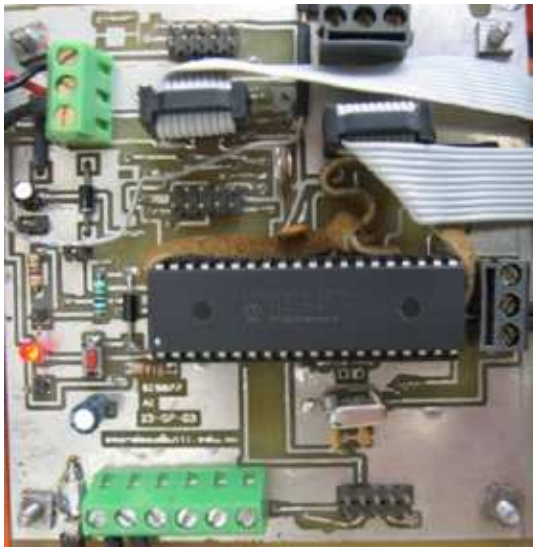

Figura A.1: Sistema mínimo basado en el PIC16F877.

Figura A.2: Autómata y circuito de prueba.

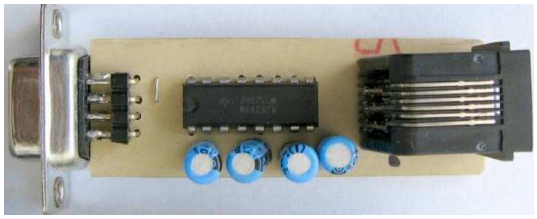

Figura A.3: Circuito de interfaz serial RS-232.

Apéndice B

Diagrama a bloques en LabVIEW

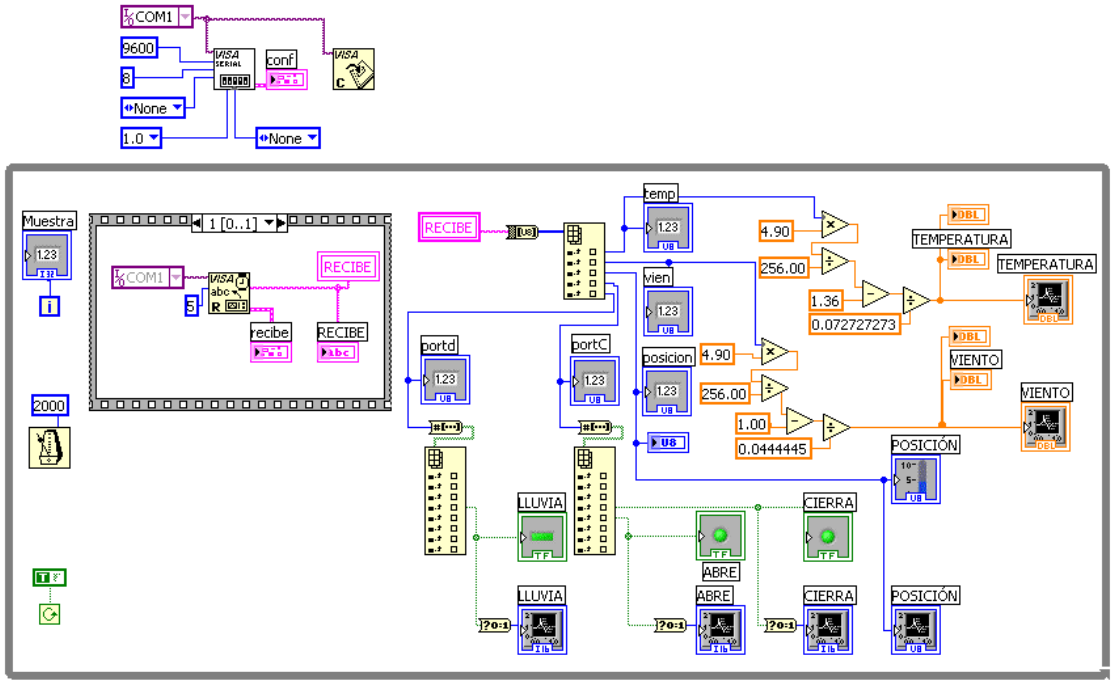

Figura B.1: Diagrama a bloques del programa en LabVIEW.