

ESCUELA POLITÉCNICA DEL EJÉRCITO

FACULTAD DE INGENIERÍA ELECTRÓNICA

PROYECTO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO
EN INGENIERÍA ELECTRÓNICA

**“SISTEMA DE MONITOREO DE ENERGÍA
EÓLICA Y SOLAR CON INTERFAZ USB”**

JUAN VLADIMIR SERRANO CEVALLOS

ANDRES ROBERTO VALLEJO LUNA

SANGOLQUÍ – ECUADOR

JULIO DEL 2006

CERTIFICACIÓN

Certificamos que el presente proyecto de grado: “Sistema de Monitoreo de Energía Eólica y Solar con Interfaz USB”, fue desarrollado en su totalidad por los señores, Juan Vladimir Serrano Cevallos y Andrés Roberto Vallejo Luna, bajo nuestra dirección

Atentamente,

Ing. Víctor Proaño

DIRECTOR

Ing. Rodolfo Gordillo

CODIRECTOR

AGRADECIMIENTO

Mis más sinceros agradecimientos en primer lugar a Dios por mantener encendida mi fe. A mis padres por haberme brindado todo su apoyo y confianza. A mi amigo Andrés por su comprensión y generosidad. Y a todos los que contribuyeron en la realización de este proyecto, en especial al Ing. Víctor Proaño, Director de Tesis, que con su amplia experiencia y conocimientos siempre supo guiarme hasta llevar a feliz término el presente Proyecto de Grado.

Juan

AGRADECIMIENTO

Yo quiero agradecer primeramente a Dios, por permitirme disfrutar de la vida. A mi esposa e hijas por ser la razón que me motiva a seguir adelante cada día. A mis padres por brindarme el apoyo y ejemplo. A mis hermanos por darme siempre lo mejor de ellos. A toda mi familia, por ayudarme en los buenos y malos momentos. A Juan por ayudarme a llevar nuestro proyecto adelante. A mis amigos que siempre me han dado su apoyo. Y al Ing. Víctor Proaño que a parte de ser un excelente profesor es una excelente persona.

Andrés

DEDICATORIA

A la memoria de mi abuelita Michita

A mis padres Juan y Ximena, a mis hermanas Andrea, Cristy y Estefy por acompañarme en todo momento y a una persona especial para mi, Alejandra.

Juan

DEDICATORIA

Dedico este trabajo a la memoria de una persona inolvidable, que con su forma de ser y cariño fue un apoyo fundamental en mi vida, Ñaño Walter.

Andrés

PRÓLOGO

El proyecto titulado “Sistema de Monitoreo de Energía Eólica y Solar con Interfaz USB” muestra el diseño, desarrollo e implementación de un sistema de sensores que permite recolectar datos de Velocidad de Viento y Radiación Solar, mediante un circuito portátil que se conecta a una PC a través del puerto USB. Finalmente se almacena la información obtenida en una base de datos.

El Sistema está compuesto por tres partes fundamentales, dos de hardware y una de software.

La primera parte del hardware está formada por un Microcontrolador PIC 16F877 acoplado a un sistema de memorias y a un RTC con comunicación I²C, que permite la recolección y almacenamiento de datos en dichas memorias. Se dispone de dos memorias seriales y el tiempo que transcurre entre la toma de datos es programable y de acuerdo a éste, se puede dimensionar la capacidad de almacenamiento del sistema.

La segunda parte del hardware está constituida por un microcontrolador PIC 16C745, el cual tiene la capacidad de comunicación USB. Este PIC está conectado al sistema de memorias con comunicación I²C y también tiene líneas de control sobre el PIC 16F877 que ayudan a que el sistema de sensores deje de tomar datos y se ponga en modo de transferencia, así mismo envía una señal al sistema de memorias para que éstas trabajen en modo solo lectura a fin de asegurar la transferencia de los datos hacia la computadora.

La programación de los Microcontroladores se la realizó utilizando el Software Micro Code Studio con librerías PICBasic Pro.

El software del sistema es un programa desarrollado en Visual Basic, que permite realizar la conexión USB con el hardware y recolectar los datos mediante un Interfaz Hombre – Máquina (HMI) y almacenarlos en una base de datos. Además se puede acceder a las tablas donde se almacena la información, se puede editar los datos guardados, visualizar gráficas de los datos en función del tiempo (fecha, hora, minuto) en que fueron tomados.

ÍNDICE

PRÓLOGO

CAPITULO I **1**

INTRODUCCIÓN A LA ENERGÍA EÓLICA Y SOLAR	1
1.1 ENERGÍA EÓLICA	1
1.1.1. Antecedentes	1
1.1.1.1. Ventajas de la Energía Eólica	2
1.1.1.2. Desventajas de la energía eólica	4
1.1.2. Transformación de la Energía	5
1.1.3. Equipos en un Sistema Eólico	5
1.1.4. Aplicaciones	8
1.1.4.1. Molinos	8
1.1.4.2. Turbinas de viento	9
1.1.4.3. Turbinas de viento modernas	10
1.1.4.4. Bombeadoras de agua	10
1.1.4.5. Generadores eléctricos	10
1.1.5. Tendencias a Futuro	12
1.2 ENERGÍA SOLAR	14
1.2.1. Antecedentes	14
1.2.1.1. Ventajas de la Energía Solar	14
1.2.1.2. Desventajas de la Energía Solar	15
1.2.2. Transformación de la Energía	16
1.2.2.1. Energía fototérmica	17
1.2.2.2. Energía fotovoltaica	17
1.2.3. Equipos de Control y Almacenamiento de la Energía	18
1.2.4. Aplicaciones	21
1.2.5. Tendencias a Futuro	22

SISTEMA DE SENSORES	24
2.1 INTRODUCCIÓN	24
2.2 IDENTIFICACIÓN DE VARIABLES A MEDIR	25
2.2.1. Velocidad del Viento	26
2.2.2. Radiación Solar	27
2.3 SISTEMA EÓLICO	28
2.3.1. Selección del Sensor	28
2.3.1.1. Transmisor de Viento – Compacto 4.3519.00.000	28
2.3.1.2. Función de Transferencia	29
2.4 SISTEMA SOLAR	29
2.4.1. Selección del Sensor	29
2.4.1.1. Standar Silizium Radiometer SSR 81	29
2.4.1.2. Circuito de Acondicionamiento	30
2.4.1.3. Función de Transferencia	31
2.5 MICROCONTROLADOR 16F877	31
2.5.1. Características Generales	32
2.5.2. Dispositivos Periféricos	33
2.5.3. Pines de Propósito General	33
2.5.4. Puertos de Entrada Salida (E/S)	33
2.5.4.1. Puerto A	34
2.5.4.2. Puerto B	34
2.5.4.3. Puerto C	34
2.5.4.4. Puerto D	35
2.5.4.5. Puerto E	35
2.5.4.6. Conversor A/D	35
2.6 ESTUDIO DE MEMORIA	37
2.6.1. Reloj Calendario de Tiempo Real	40
2.6.2. Modo de Almacenamiento de Datos	42
2.7 TARJETA DE ALMACENAMIENTO DE DATOS	44
2.7.1. Código del Circuito de Almacenamiento De Datos	44
2.7.1.1. Introducción a MicroCode Studio	44

2.7.1.2. Diagrama de Flujo del Código del Programa	48
2.7.2. Diagrama del Circuito de Almacenamiento de Datos	50
2.8 DIAGRAMA DE BLOQUES DEL SISTEMA DE SENSORES	50

CAPITULO III **51**

DISEÑO DEL SISTEMA DE MONITOREO Y ANÁLISIS EÓLICO - SOLAR	51
3.1 TARJETA USB DE ADQUISICIÓN DE DATOS	51
3.1.1. Introducción al Puerto USB	51
3.1.2. Universal Serial Bus (USB)	52
3.1.2.1. ¿Qué es USB?	52
3.1.2.2. ¿Cómo funciona?	53
3.1.2.3. Controlador	54
3.1.2.4. Concentradores o Hubs	54
3.1.2.5. Periféricos	55
3.1.2.6. Diagrama de capas	56
3.1.2.7. Cables y conectores	58
3.1.3. Aspectos Básicos del Firmware para el Microcontrolador	59
3.1.4. Diseño del Interfaz USB	61
3.1.4.1. Código del Programa de la Tarjeta de Adquisición USB	63
3.1.4.2. Diagrama del Circuito de la Tarjeta de Adquisición USB	63
3.2 DIAGRAMA DE BLOQUES DEL SISTEMA COMPLETO	63
3.2.1. Diagrama Electrónico del Circuito Total	65
3.2.2. Diseño de la Placa del Circuito (PCB)	65
3.3 DESARROLLO DEL HMI DE OPERACIÓN DEL SISTEMA	65
3.3.1. Desarrollo del Código de la Aplicación HMI	67
3.3.2. Programación de la Aplicación HMI	68
3.3.3. Presentación del Interfaz Gráfico del Programa	72
3.4 ANÁLISIS DE RESULTADOS	76

CAPITULO IV **80**

CONCLUSIONES Y RECOMENDACIONES 80

4.1 CONCLUSIONES 80

4.2 RECOMENDACIONES 81

REFERENCIAS BIBLIOGRÁFICAS **83**

ANEXOS **84**

ANEXO A.	HOJA DE DATOS TÉCNICA DEL SENSOR DE VELOCIDAD DEL VIENTO	85
ANEXO B.	HOJA DE DATOS TÉCNICA DEL SENSOR DE RADIACIÓN SOLAR	89
ANEXO C.	HOJA DE DATOS TÉCNICA DE LA MEMORIA 24LC256	92
ANEXO D.	HOJA DE DATOS TÉCNICA DEL RTC	97
ANEXO E.	CÓDIGO DEL PROGRAMA DEL SISTEMA DE SENSORES	102
ANEXO F.	DIAGRAMA DEL CIRCUITO DE SENSORES ACOPLADO AL CIRCUITO DE ALMACENAMIENTO DE DATOS	111
ANEXO G.	HOJA DE DATOS TÉCNICA DEL MICROCONTROLADOR 16C745	113
ANEXO H.	USB FIRMWARE DE MICROCHIP	121
ANEXO I.	CÓDIGO DEL PROGRAMA PARA LA COMUNICACIÓN DE DATOS USB	130
ANEXO J.	DIAGRAMA DEL CIRCUITO DE MONITOREO DE ENERGÍA EÓLICA Y SOLAR CON INTERFAZ USB	134
ANEXO K.	DIAGRAMA DE LAS PLACAS DEL CIRCUITO DE MONITOREO DE ENERGÍA EÓLICA Y SOLAR CON INTERFAZ USB	136
ANEXO L.	CÓDIGO DEL PROGRAMA DE LA INTERFAZ HOMBRE - MÁQUINA (HMI)	138
ANEXO M.	ANÁLISIS DE COSTOS	164

ÍNDICE DE FIGURAS **166**

ÍNDICE DE TABLAS **167**

GLOSARIO **168**

CAPITULO I

INTRODUCCIÓN A LA ENERGÍA EÓLICA Y SOLAR

1.1 ENERGÍA EÓLICA

1.1.1. Antecedentes

La energía eólica es la energía producida por el viento, pero es una variable de la energía solar, pues se deriva del calentamiento diferencial de la atmósfera y de las irregularidades de relieve de la superficie terrestre.

Durante el día el sol calienta el aire sobre tierra firme más que el que está sobre el mar. El aire continental se expande y eleva, disminuyendo así la presión sobre el terreno y haciendo que el viento sople desde el mar hacia las costas. La rotación terrestre, la diferencia de temperatura y la presión atmosférica tienen influencia en la dirección del viento. El contenido energético del viento depende de su velocidad. Cerca del suelo, la velocidad es baja, pero aumenta rápidamente con la altura. Cuanto más accidentada sea la superficie del terreno, más frenará ésta al viento. El viento sopla con menos velocidad en las depresiones terrestres y con mayor velocidad sobre las colinas, pero en grandes valles y terreno montañoso nos encontramos con el efecto túnel que puede proporcionar buenas velocidades de viento. No obstante, el viento sopla con más fuerza sobre el mar que en tierra. Es por esto, que las mejores localizaciones para su aprovechamiento se encuentren en el mar, sobre colinas, cercanas a la costa y con poca vegetación.

La primera utilización de la capacidad energética del viento la constituye la navegación a vela. En ella, la fuerza del viento se utiliza para impulsar un barco. Barcos con velas aparecían ya en los grabados egipcios más antiguos (3 000 a.C.). Los egipcios, los fenicios y más tarde los romanos tenían que utilizar también los remos para contrarrestar una característica esencial de la energía eólica, su discontinuidad.

Efectivamente, el viento cambia de intensidad y de dirección de manera impredecible, por lo que había que utilizar los remos en los períodos de calma o cuando no soplaba en la dirección deseada. Hoy, en los parques eólicos, se utilizan los acumuladores para producir electricidad durante un tiempo, cuando el viento no sopla.

Otra característica de la energía producida por el viento es su infinita disponibilidad en función lineal a la superficie expuesta a su incidencia. En los barcos, a mayor superficie de sus velas mayor velocidad. En los parques eólicos, cuantos más molinos haya, más potencia en bornes de la central. En los veleros, el aumento de superficie de las velas tiene limitaciones mecánicas (se rompe el mástil o vuelca el barco). En los parques eólicos las únicas limitaciones al aumento del número de molinos son las urbanísticas.

1.1.1.1. Ventajas de la Energía Eólica

La energía eólica no contamina, es inagotable y frena el agotamiento de combustibles fósiles contribuyendo a evitar el cambio climático. Es una tecnología de aprovechamiento totalmente madura y puesta a punto.

Es una de las fuentes más baratas, puede competir en rentabilidad con otras fuentes energéticas tradicionales como las centrales térmicas de carbón (considerado tradicionalmente como el combustible más barato), las centrales de combustible e incluso con la energía nuclear, si se consideran los costes de reparar los daños medioambientales.

El generar energía eléctrica sin que exista un proceso de combustión o una etapa de transformación térmica supone, desde el punto de vista medioambiental, un procedimiento muy favorable por ser limpio, exento de problemas de contaminación, etc. Se suprimen radicalmente los impactos originados por los combustibles durante su extracción, transformación, transporte y combustión, lo que beneficia la atmósfera, el suelo, el agua, la fauna, la vegetación, etc.

Evita la contaminación que conlleva el transporte de los combustibles; gas, petróleo, gasoil, carbón. Reduce el intenso tráfico marítimo y terrestre cerca de las centrales. Suprime los riesgos de accidentes durante estos transportes: desastres con petroleros

(traslados de residuos nucleares, etc.). No hace necesaria la instalación de líneas de abastecimiento: canalizaciones a las refinerías o las centrales de gas.

La utilización de la energía eólica para la generación de electricidad presenta nula incidencia sobre las características fisicoquímicas del suelo o su erosión, ya que no se produce ningún contaminante que incida sobre este medio, ni tampoco vertidos o grandes movimientos de tierras.

La generación de electricidad a partir del viento no produce gases tóxicos, ni contribuye al efecto invernadero, ni destruye la capa de ozono, tampoco crea lluvia ácida. No origina productos secundarios peligrosos ni residuos contaminantes. Cada KW/h de electricidad generada por energía eólica en lugar de carbón, evita:

0.60 Kg. de CO₂, dióxido de carbono

1.33 gr. de SO₂, dióxido de azufre

1.67 gr. de NO_x, óxido de nitrógeno

La electricidad producida por un aerogenerador evita que se quemen diariamente miles de litros de petróleo en las centrales térmicas. Ese mismo generador produce idéntica cantidad de energía que la obtenida por quemar diariamente 1 000 Kg. de petróleo. Al no quemarse esos Kg. de carbón, se evita la emisión de 4 109 Kg. de CO₂, lográndose un efecto similar al producido por 200 árboles. Se impide la emisión de 66 Kg. de dióxido de azufre -SO₂- y de 10 Kg. de óxido de nitrógeno -NO_x- principales causantes de la lluvia ácida.

La energía eólica es independiente de cualquier política o relación comercial, se obtiene en forma mecánica y por tanto es directamente utilizable. En cuanto a su transformación en electricidad, ésta se realiza con un rendimiento excelente y no a través de aparatos termodinámicos con un rendimiento de Carnot (de energía) siempre pequeño. Al finalizar la vida útil de la instalación, el desmantelamiento no deja huellas.

Tabla. 1.1. Ventajas en un campo de 10 MW

Evita	28 480 toneladas al año de CO ₂
Sustituye	2 447 toneladas equivalentes de petróleo
Aporta	Trabajo a 130 personas al año apróx. durante el diseño y la construcción
Proporciona	Industria y desarrollo de tecnología
Genera	Energía eléctrica para 11 000 familias apróx.

1.1.1.2. Desventajas de la energía eólica

El aire al ser un fluido de pequeño peso específico, implica la fabricación de máquinas grandes y en consecuencia caras para generar energía. La altura de estas máquinas puede igualar a la de un edificio de diez o más plantas, en tanto que la envergadura total de sus aspas alcanza la veintena de metros, lo cual encarece su producción.

Desde el punto de vista estético, la energía eólica produce un impacto visual inevitable, ya que por sus características precisa unos emplazamientos que normalmente resultan ser los que más evidencian la presencia de las máquinas (cerros, colinas, litoral). En este sentido, la implantación de la energía eólica a gran escala, puede producir una alteración clara sobre el paisaje, que deberá ser evaluada en función de la situación previa existente en cada localización.

Un impacto negativo es el ruido producido por el giro del rotor, pero su efecto no es más delatado que el generado por una instalación de tipo industrial de similar capacidad. El ruido será más notorio dependiendo de la proximidad a la que nos encontremos con respecto a los aerogeneradores.

También ha de tenerse especial cuidado a la hora de seleccionar un parque si en las inmediaciones habitan aves, por el riesgo de mortandad al impactar con las palas, aunque existen soluciones al respecto como pintar en colores llamativos las palas, situar los molinos adecuadamente dejando “pasillos” a las aves, e incluso en casos extremos hacer un seguimiento de las aves por radar llegando a parar las turbinas para evitar las colisiones.

1.1.2. Transformación de la Energía

La energía del viento es de tipo cinético (debida a su movimiento); ello hace que la potencia obtenible del mismo dependa de forma acusada de su velocidad, así como del área de la superficie captadora. Así, todas las máquinas que ha construido el hombre para obtener el mayor rendimiento posible de la energía del viento se basan en frenar el viento por medio de algún dispositivo colocado en su camino.

La energía del viento se ha utilizado desde tiempos antiguos y actualmente se aprovecha para la producción de electricidad. Una máquina eólica consta de un rotor o hélice colocada a determinada altura, que se mueve por el viento y está conectado a un aerogenerador de electricidad.

Una máquina eólica es cualquier dispositivo accionado por el viento. Si se utiliza directamente la energía mecánica, será un aeromotor, y si se acciona un generador eléctrico, se tratará de un aerogenerador.

1.1.3. Equipos en un Sistema Eólico

Los elementos que encontramos en un sistema eólico son los siguientes:

- Soportes
- Sistema de captación
- Sistema de orientación
- Sistema de regulación
- Sistema de transmisión
- Sistema de generación

Las máquinas eólicas se han de colocar sobre un soporte que resista el empuje del viento y que permita elevarla lo suficiente para evitar las turbulencias debidas al suelo o a los obstáculos cercanos. Se pueden distinguir dos tipos de soportes:

Autoportantes: torres de estructura metálica, de hormigón o tubulares.

Atirantados: estructuras unidas al suelo por cables tensados que permiten, en las máquinas pequeñas, abatir la máquina para su mantenimiento o reparación.

El sistema de captación o "rotor" es el elemento principal de una máquina eólica. Está compuesto por cierto número de "palas" y su misión es transformar la energía del viento en energía mecánica utilizable. Existe gran variedad de rotores, tanto de eje horizontal como de eje vertical.

Los parámetros principales de un rotor son los siguientes:

Velocidad típica de giro: relación entre la velocidad de la punta de la pala y la velocidad del viento, permite su clasificación en rotores lentos (velocidad típica próxima a 1 m/s) o rápidos (velocidad típica de 5 a 8 m/s).

Rendimiento aerodinámico: también llamado "coeficiente de potencia", expresa la fracción de la energía del viento que se transforma en energía mecánica; su valor oscila entre el 20 y el 40 %, dependiendo de las siguientes características geométricas de las palas:

- Longitud
- Perfil, o forma del borde de ataque de la pala contra el viento
- Calaje, o ángulo de ataque de la pala contra el viento
- Anchura

En los últimos años se han ensayado tanto la madera, diversas aleaciones metálicas y polímeros de resinas plásticas, habiendo dado estos últimos muy buenos resultados, con una considerable reducción de los costes de la pala.

Los sistemas de orientación sólo son necesarios en las máquinas eólicas de eje horizontal y constan de un mecanismo que detecta la dirección del viento y sitúa el rotor en su misma dirección.

Para máquinas de pequeña y mediana potencia (menor de 50 KW.), cuyo rotor está situado cara al viento, el dispositivo más adecuado suele ser una cola, superficie plana situada en el extremo de un soporte unido al cuerpo del aeromotor. También se utilizan rotores auxiliares dos pequeñas hélices tras el rotor y en dirección perpendicular al mismo, sobre las que sólo actúa el viento cuando el rotor no está orientado.

Las máquinas mayores, cuyo rotor suele estar situado detrás de la torre, utilizan para su orientación el efecto de conicidad, basado en una leve inclinación de las palas hacia atrás. Finalmente, en las grandes máquinas, se dispone de motores auxiliares que funcionan de forma automática orientando el rotor en la dirección adecuada.

Los sistemas de regulación tienen por objeto controlar la velocidad de rotación, evitando las fluctuaciones producidas por la velocidad del viento. Los sistemas más sencillos operan sólo con vientos muy fuertes, frenando el rotor; los más elaborados adaptan la máquina a cualquier condición de viento y de potencia.

El sistema de regulación más sencillo es el de "puesta en bandera", que produce un frenado al situar el rotor paralelo al viento; el rotor es devuelto a su posición normal cuando la velocidad del viento disminuye.

El sistema de paso variable es probablemente la forma de regulación más eficaz y actúa variando el ángulo de ataque de las palas, lo que hace variar el rendimiento aerodinámico y, en consecuencia, la potencia absorbida. Esto se lleva a cabo con ayuda de diversos mecanismos, que varían según el tamaño y características de la máquina eólica y no sólo se puede regular la velocidad de giro de la máquina actuando sobre el rotor; también se puede actuar sobre el eje motor, realizándose el control de la potencia mediante el frenado del mismo. El freno puede ser de zapatas, de disco o de tipo electromagnético y puede actuar por distintos mecanismos. Aunque este sistema tiene que realizar esfuerzos mayores, posee la ventaja de ser más sencillo y de encontrarse ya comercializado, hecho que disminuye considerablemente sus costes.

Para poder aprovechar la energía mecánica obtenida en el rotor es necesario un sistema de transmisión. La energía mecánica se puede transmitir como tal o mediante poleas, engranajes o utilizando un sistema cigüeñal-biela. Cuando se desea generar energía eléctrica es necesario primero aumentar la velocidad de giro del rotor, que suele ser inferior a las 200 r.p.m. antes de accionar el generador. Ello se logra con un multiplicador, que puede ser de diferentes tipos. Se logran así velocidades de giro de varios miles de revoluciones por minuto, que permiten accionar generadores eléctricos.

Los sistemas de generación se utilizan en las máquinas eólicas para producir energía eléctrica, forma de energía preferible a la mecánica, debido a la facilidad en su manipulación y transporte así como a la versatilidad que presentan sus aplicaciones posteriores. El sistema eléctrico de un aerogenerador está condicionado por la velocidad de operación del rotor y por el uso que se le dé a la energía obtenida. Los generadores que transforman la energía mecánica en eléctrica pueden ser dínamos o alternadores.

La dínamo es una máquina eléctrica sencilla que produce corriente continua y es capaz de proporcionar potencia a escasas revoluciones, con lo que a veces se puede prescindir de la etapa previa de multiplicación.

Normalmente, la corriente eléctrica producida por una dínamo se almacena como tal en un sistema de acumulación de baterías.

Los alternadores generan corriente alterna y para una misma potencia son más ligeros, baratos y de mayor rendimiento que las dínamos, pero trabajan a elevado número de revoluciones, por lo que necesitan un multiplicador.

1.1.4. Aplicaciones

1.1.4.1. Molinos

Los molinos movidos por el viento tienen un origen remoto. En el siglo VII d.C. ya se utilizaban molinos elementales en Persia (hoy, Irán) para el riego y moler el grano. En estos primeros molinos la rueda que sujetaba las aspas era horizontal y estaba soportada sobre un eje vertical. Estas máquinas no resultaban demasiado eficaces, pero aún así se extendieron por China y el Oriente Próximo.

En Europa los primeros molinos aparecieron en el siglo XII en Francia e Inglaterra y se distribuyeron por el continente. Eran unas estructuras de madera, conocidas como torres de molino, que se giraban a mano alrededor de un poste central para levantar sus aspas al viento.

El molino de torre se desarrolló en Francia a lo largo del siglo XIV. Consistía en una torre de piedra coronada por una estructura rotativa de madera que soportaba el eje del molino y la maquinaria superior del mismo.

Estos primeros ejemplares tenían una serie de características comunes. De la parte superior del molino sobresalía un eje horizontal. De este eje partían de cuatro a ocho aspas, con una longitud entre 3 y 9 metros. Las vigas de madera se cubrían con telas o planchas de madera. La energía generada por el giro del eje se transmitía, a través de un sistema de engranajes, a la maquinaria del molino emplazada en la base de la estructura.

Además de emplearse para el riego y moler el grano, los molinos construidos entre los siglos XV y XIX tenían otras aplicaciones, como el bombeo de agua en tierras bajo el nivel del mar, aserraderos de madera, fábricas de papel, prensado de semillas, para producir aceite, así como para triturar todo tipo de materiales. En el siglo XIX se llegaron a construir unos 9 000 molinos en Holanda.

El avance más importante fue la introducción del abanico de aspas, inventado en 1745, que giraba impulsado por el viento. En 1772 se introdujo el aspa con resortes. Este tipo de aspa consiste en unas cerraduras de madera que se controlan de forma manual o automática, a fin de mantener una velocidad de giro constante en caso de vientos variables. Otros avances importantes han sido los frenos hidráulicos para detener el movimiento de las aspas y la utilización de aspas aerodinámicas en forma de hélice, que incrementan el rendimiento de los molinos con vientos débiles.

1.1.4.2. Turbinas de viento

El uso de las turbinas de viento para generar electricidad comenzó en Dinamarca a finales del siglo pasado y se ha extendido por todo el mundo. Los molinos para el bombeo de agua se emplearon a gran escala durante el asentamiento en las regiones áridas del oeste de Estados Unidos. Pequeñas turbinas de viento generadoras de electricidad abastecían a numerosas comunidades rurales hasta la década de los años treinta, cuando en Estados Unidos se extendieron las redes eléctricas. También se construyeron grandes turbinas de viento en esta época.

1.1.4.3. Turbinas de viento modernas

Las modernas turbinas de viento se mueven por dos procedimientos: el arrastre, en el que el viento empuja las aspas, y la elevación, en el que las aspas se mueven de un modo parecido a las alas de un avión a través de una corriente de aire. Las turbinas que funcionan por elevación giran a más velocidad y son, por su diseño, más eficaces. Las turbinas de viento pueden clasificarse en turbinas de eje horizontal, en las que los ejes principales están paralelos al suelo y turbinas de eje vertical, con los ejes perpendiculares al suelo. Las turbinas de ejes horizontales utilizadas para generar electricidad tienen de una a tres aspas, mientras que las empleadas para bombeo pueden tener muchas más. Entre las máquinas de eje vertical más usuales destacan las Savonius, cuyo nombre proviene de sus diseñadores, y que se emplean sobre todo para bombeo; y las Darrieus, una máquina de alta velocidad que se asemeja a una batidora de huevos.

1.1.4.4. Bombeadoras de agua

Una bombeadora de agua es un molino con un elevado momento de torsión y de baja velocidad, frecuente en las regiones rurales de Estados Unidos. Las bombeadoras de agua se emplean sobre todo para drenar agua del subsuelo. Estas máquinas se valen de una pieza rotatoria, cuyo diámetro suele oscilar entre 2 y 5 m. con varias aspas oblicuas que parten de un eje horizontal. La pieza rotatoria se instala sobre una torre lo bastante alta como para alcanzar el viento. Una larga veleta en forma de timón dirige la rueda hacia el viento. La rueda hace girar los engranajes que activan una bomba de pistón. Cuando los vientos arrecian en exceso, unos mecanismos de seguridad detienen de forma automática la pieza rotatoria para evitar daños en el mecanismo.

1.1.4.5. Generadores eléctricos

Los científicos calculan que hasta un 10% de la electricidad mundial se podría obtener de generadores de energía eólica a mediados del siglo XXI. Los generadores de turbina de viento tienen varios componentes. El rotor convierte la fuerza del viento en energía rotatoria del eje, una caja de engranajes aumenta la velocidad y un generador transforma la energía del eje en energía eléctrica.

En algunas máquinas de eje horizontal la velocidad de las aspas puede ajustarse y regularse durante su funcionamiento normal, así como cerrarse en caso de viento excesivo. Otras emplean un freno aerodinámico que con vientos fuertes reduce automáticamente la energía producida. Las máquinas modernas comienzan a funcionar cuando el viento alcanza una velocidad de unos 19 Km/h, logran su máximo rendimiento con vientos entre 40 y 48 Km/h y dejan de funcionar cuando los vientos alcanzan los 100 Km/h. Los lugares ideales para la instalación de los generadores de turbinas son aquellos en los que el promedio anual de la velocidad del viento es de cuando menos 21 Km/h.

La energía eólica, que no contamina el medio ambiente con gases ni agrava el efecto invernadero, es una valiosa alternativa frente a los combustibles no renovables como el petróleo. Los generadores de turbinas de viento para producción de energía a gran escala y de rendimiento satisfactorio tienen un tamaño mediano (de 15 a 30 metros de diámetro, con una potencia entre 100 y 400 KW). Algunas veces se instalan en filas y se conocen entonces como granjas de viento (Fig. 1.1). En California se encuentran algunas de las mayores granjas de viento del mundo y sus turbinas pueden generar unos 1 120 MW de potencia (una central nuclear puede generar unos 1 100 MW).

El precio de la energía eléctrica producida por ese medio resulta competitivo con otras muchas formas de generación de energía. En la actualidad Dinamarca obtiene más del 2% de su electricidad de las turbinas de viento. Este tipo de generación es también empleada para aumentar el suministro de electricidad a comunidades insulares y en lugares remotos.

En Gran Bretaña, uno de los países más ventosos del mundo, los proyectos de turbinas de viento, especialmente en Gales y en el noroeste de Inglaterra, generan una pequeña parte de la electricidad procedente de fuentes de energía renovable.

En España se inauguró en el año 1986 un parque eólico de gran potencia en Tenerife, Canarias. Más tarde se hicieron otras instalaciones en La Muela (Zaragoza), el Ampurdán (Gerona), Estaca de Bares (La Coruña) y Tarifa (Cádiz), ésta dedicada fundamentalmente a la investigación. La energía eólica supone un 6% de la producción de energía primaria en los países de la Unión Europea.



Figura. 1.1. Generadores de Energía Eólica

1.1.5. Tendencias a Futuro

La Energía Eólica está experimentando en la actualidad una gran expansión, sobre todo en Europa. Constituye una alternativa a otras fuentes de energía como son la energía nuclear, térmica, etc. todas ellas altamente contaminantes.

Se trata de una tecnología muy antigua que ha vivido su mayor crecimiento a partir de la década de los 70 (debido a la crisis del petróleo del 73), y especialmente en la última década. Este crecimiento tan espectacular se debe fundamentalmente a la mejora del rendimiento energético y a una mayor potencia de generación, tanto en pequeños como en grandes sistemas eólicos, lo que redundará en un acercamiento entre la tecnología eólica y los sistemas de generación convencionales.

La energía eólica aprovecha una fuente de energía infinita, inagotable y totalmente gratis como es el viento. A modo de ejemplo: la energía cinética contenida en el viento, a escala mundial, es 80 veces mayor al consumo energético de toda la humanidad.

Una de las mayores preocupaciones de los países industrializados es el miedo al agotamiento de las fuentes tradicionales: carbón, petróleo, gas natural, etc. El ritmo de

consumo de estas materias primas es tal, que en un año la humanidad consume lo que la naturaleza ha tardado un millón de años en producir. Por tanto no es descabellado pensar que las reservas se agotarán en años futuros. La energía eólica aliviará esta preocupación.

Otro factor importante es el carácter no contaminante de esta fuente de energía: se suprimen radicalmente los impactos originados por los combustibles durante su extracción, transformación, transporte y combustión, lo que incide beneficiosamente en la atmósfera, el suelo, el agua, ya que las plantas eólicas evitan el efecto invernadero y la lluvia ácida.

Por último, el factor económico. Las instalaciones eólicas presentan una rápida amortización, siendo para las grandes centrales 3 ó 5 meses. Además, a lo largo de su vida (unos 25 años) generan unas 30 veces más la energía que se utilizó para su construcción, término que se conoce como "balance de energía" de los sistemas eólicos. En la actualidad, el precio de la electricidad generada mediante la energía eólica está en torno a los 5 cent/KWh, un precio muy competitivo comparado con las restantes formas de generar energía, lo que promocionaría en un futuro el fortalecimiento de este tipo de energía.

En Ecuador se tiene un gran potencial para la instalación de parques eólicos, de esta forma se generaría electricidad y así poder suplir la demanda de energía en momentos de estiaje.

Loja sería el punto de partida en nuestro país para una producción de energía eléctrica a mediana escala, ya que si bien encontramos generadores eólicos en algunas zonas de nuestro territorio, solamente se los usa para autoabastecimiento. El Proyecto de un parque eólico en Loja se lo llevaría a cabo gracias a la colaboración del Ministerio de Energía y Minas, el Consejo Municipal de Loja y Apoyo del Gobierno Español.

1.2 ENERGÍA SOLAR

1.2.1. Antecedentes

La energía solar es la energía radiante producida en el sol como resultado de reacciones nucleares de fusión. Llega a la Tierra a través del espacio en cuantos de energía llamados fotones, que interactúan con la atmósfera y la superficie terrestres.

El sol siempre ha sido una fuente de energía. Las plantas utilizan la luz del sol para realizar la fotosíntesis y nutrirse. Los animales se comen estas plantas y las plantas y animales de hace millones de años son el petróleo de hoy en día. Así que podemos decir que los combustibles fósiles se iniciaron con la energía del sol.

Hoy en día, muchas casas, principalmente en los países desarrollados, utilizan la energía del sol para calentar sus hogares, la estufa y el agua que emplean. La energía del sol se capta con unos paneles formados por espejos localizados en los techos de edificios y casas. Los rayos luminosos se concentran con los espejos, los cuales mandan esta energía ya sea para calentar tuberías de agua o bien, para obtener electricidad.

1.2.1.1. Ventajas de la Energía Solar

La energía solar, además de ser renovable y no contaminar el Medio Ambiente, es una energía muy abundante. Su utilización contribuye a reducir el efecto invernadero producido por las emisiones de CO₂ a la atmósfera, así como el cambio climático provocado por el efecto invernadero. Además, con su difusión y promoción todos colaboramos a que en el futuro se aproveche también el sol en escuelas y edificios.

Entre las ventajas medioambientales tenemos la disminución de las emisiones de CO₂. Por cada 20 KWh de electricidad producida a partir de energía solar se dejan de emitir unos 10 Kg. de CO₂ al año, en 25 años se evitan 250 Kg. de CO₂. Reducción de la contaminación atmosférica, del efecto invernadero producido por las emisiones de CO₂ y del cambio climático provocado por el efecto invernadero.

En cuanto a los beneficios educativos, está la educación de los alumnos en las tecnologías ecológicas y en la cultura de respeto al Medio Ambiente. La posibilidad de creación de un Club Solar en cada centro, que recoja, estudie y elabore trabajos sobre temas relacionados con la energía solar y la programación de actividades educativas relacionadas con la energía solar y la ecología.

Los beneficios económicos⁽¹⁾ son muchos, cada kilovatio-hora (KWh) producido con energía solar fotovoltaica se cobra a \$0.48. Una instalación de 5 KW de potencia puede producir al año entre 5 000 y 7 500 KWh, es decir, entre \$2 500 y \$3 700. Una instalación de 5 KW de potencia cuesta aproximadamente unos \$37 000 o \$44 000 y puede producir entre 150 000 y 187 500 KWh en 25 a 30 años, es decir de \$73 000 a \$87 000.

Entre los beneficios sociales constan que las energías renovables generan más puestos de trabajo que otras energías más contaminantes. Por cada \$600 000 invertidos en energía solar se crean entre 4 y 6 nuevos empleos. La misma inversión en energía procedente del petróleo sólo crearía 0.6 puestos de trabajo.

Los puestos generados por la inversión en energía solar no son estacionarios (ligados a la construcción de una central, etc.), y se distribuyen a pequeña escala por todo el territorio.

La utilización de energía solar en zonas rurales o aisladas, permite la creación de pequeñas empresas, lo que potencia el desarrollo económico de comarcas poco favorecidas.

1.2.1.2. Desventajas de la Energía Solar

Se puede decir que la mayor desventaja que presenta la energía solar es su costo de inversión inicial. Siendo este factor el que impide que se desarrollen más proyectos de este tipo en nuestro país.

(1) los beneficios económicos son tomados de países desarrollados

1.2.2. Transformación de la Energía

La recogida natural de energía solar se produce en la atmósfera, los océanos y las plantas de la Tierra. Las interacciones de la energía del sol, los océanos y la atmósfera, por ejemplo, producen vientos, utilizados durante siglos para hacer girar los molinos.

Casi el 30% de la energía solar que alcanza el borde exterior de la atmósfera se consume en el ciclo del agua, que produce la lluvia y la energía potencial de las corrientes de montañas y de los ríos. La energía que generan estas aguas en movimiento al pasar por las turbinas modernas se llama energía hidroeléctrica.

Gracias al proceso de fotosíntesis, la energía solar contribuye al crecimiento de la vida vegetal (biomasa) que, junto con la madera y los combustibles fósiles que desde el punto de vista geológico derivan de plantas antiguas, puede ser utilizada como combustible. Otros combustibles como el alcohol y el metano también pueden extraerse de la biomasa.

Asimismo, los océanos representan un tipo natural de recogida de energía solar. Como resultado de su absorción por los océanos y por las corrientes oceánicas, se producen gradientes de temperatura. En algunos lugares, estas variaciones verticales alcanzan 20 °C en distancias de algunos cientos de metros. La diferencia entre estas energías se manifiesta como energía mecánica (para mover una turbina, por ejemplo), que puede conectarse a un generador, para producir electricidad. Estos sistemas, llamados sistemas de conversión de energía térmica oceánica (CETO), requieren enormes intercambiadores de energía y otros aparatos en el océano para producir potencias del orden de megavatios.

Existen dos tipos de celdas convertidoras de energía solar, una es la fototérmica y la otra la fotovoltaica:

1.2.2.1. Energía fototérmica

Los sistemas fototérmicos convierten la radiación solar en calor y lo transfieren a un fluido de trabajo. El calor se usa para calentar agua, edificios, mover turbinas para generar electricidad, secar granos o destruir desechos peligrosos.

Los colectores térmicos solares se dividen en:

los de baja temperatura (proveen calor útil de temperaturas menores de 65 °C , se usan para calentar albercas, pasteurización de leche, lavados textiles y agua para baño)

los de temperatura media (controlan la radiación solar para entregar calor útil a temperaturas entre 100 y 300 °C, utilizan espejos que concentran los rayos)

los de alta temperatura (operan a temperaturas mayores de 500 °C y se usan para generar electricidad)

1.2.2.2. Energía fotovoltaica

Los sistemas fotovoltaicos convierten directamente parte de la energía solar en electricidad. Las celdas fotovoltaicas se fabrican principalmente de silicio (el segundo elemento más abundante en la corteza terrestre).

Los electrones de la celda son excitados por la luz solar y se mueven a través del silicio, produciendo el efecto fotovoltaico, el cual genera una corriente eléctrica.

Existen celdas fotovoltaicas en nuestras calculadoras solares así como en los cohetes espaciales. Hoy en día existen muchos proyectos para colocarlas en vehículos que utilicen energía solar para transportarse.

Esta tecnología no se ha desarrollado por completo en Ecuador, y lo que se quiere es realizar mapas de irradiación del país, es decir, saber cuales son las zonas de Ecuador a las cuales les llegan más rayos de sol. En la actualidad ya existen varias celdas fotovoltaicas instaladas en distintos lugares del país.

En estos sistemas muchas celdas son colocadas en placas planas o concentradores para aumentar su eficiencia, teóricamente su capacidad de generación es limitada sin

embargo a la actualidad, el rango es de pocos kilovatios de potencia eléctrica (KWe) con un máximo de 5 MWe. La eficiencia actual está entre 5% y 15%, mientras que los módulos concentradores tienen rendimientos próximos al 20%. El impacto en el medio ambiente se ha visto afectado por las grandes fábricas productoras de celdas fotovoltaicas que generan gases tóxicos, líquidos y sustancias sólidas dañinas para la salud.

1.2.3. Equipos de Control y Almacenamiento de la Energía

Los componentes de un sistema fotovoltaico dependen del tipo de aplicación que se considere (autónoma o conectada a la red) y de las características de la instalación.

Para el caso de un sistema autónomo, los componentes necesarios para que la instalación funcione correctamente y tenga una elevada fiabilidad de suministro y durabilidad son: placas fotovoltaicas, acumulador eléctrico, regulador de carga e inversor.

Por su lado, las instalaciones conectadas a la red de distribución eléctrica se caracterizan por no incorporar acumuladores, ya que la energía que se envía a la red no necesita acumularse.

El principal elemento de una instalación fotovoltaica es la placa solar fotovoltaica. Estas placas son las que producen electricidad en forma de corriente continua. Para optimizar su rendimiento hace falta orientar las placas en dirección sur con una inclinación determinada que depende del día del mes etc. La utilización de acumuladores está motivada por el hecho que la intensidad solar varía a lo largo del día y del año, mientras que las necesidades energéticas no lo hacen de manera paralela a estas fluctuaciones y por tanto, hay que guardarla. Normalmente se utilizan acumuladores eléctricos, ya que es el sistema más eficiente y económico del que se dispone. Las instalaciones fotovoltaicas utilizan normalmente baterías estacionarias con largos períodos de descarga que se adaptan mejor al régimen de funcionamiento de dichas instalaciones.

La capacidad de acumular la electricidad necesaria se calcula realizando una media de necesidades y de días de autonomía necesarios, normalmente unas 100 horas. Siempre hay el riesgo de quedarse sin energía después de largos días de mal tiempo.

Las células fotovoltaicas suelen tener un grosor de entre 200 y 400 micras y una superficie de 8 a 10 cm². En lo que se refiere a la potencia que proporcionan los sistemas fotovoltaicos, una misma célula proporciona valores diferentes al variar la intensidad de radiación que recibe. Por éste motivo, normalmente la potencia nominal de las células se mide en pico Vatios, que es la potencia que puede proporcionar la célula, con una intensidad de radiación constante de 1 000 W/m² y a una temperatura de la célula de 25°. Por ejemplo, una instalación de 10 Wp (Vatio pico, unidad de energía fotovoltaica), suministra una potencia de 10 W cuando se le irradia 1 000 W/m². En general una célula individual tiene una potencia nominal de 1Wp, lo que quiere decir que bajo esa radiación proporciona valores de tensión de unos 0.5 V y corrientes de 2 A.

Para obtener potencias utilizables por equipos eléctricos de mediana potencia, hace falta unir un cierto número de células con la finalidad de obtener la tensión y la corriente requeridas. La unidad básica de las instalaciones fotovoltaicas es, pues, la placa fotovoltaica, que suele contener entre 20 y 40 células solares para producir corriente continua de 10 o 24 voltios y proporcionar valores de potencia que oscilen entre 50 y 100 Wp. Normalmente las células van encapsuladas en silicona y cerradas con una carcasa periférica metálica. Para obtener otros valores de tensión y potencia las placas se pueden conectar en serie o en paralelo

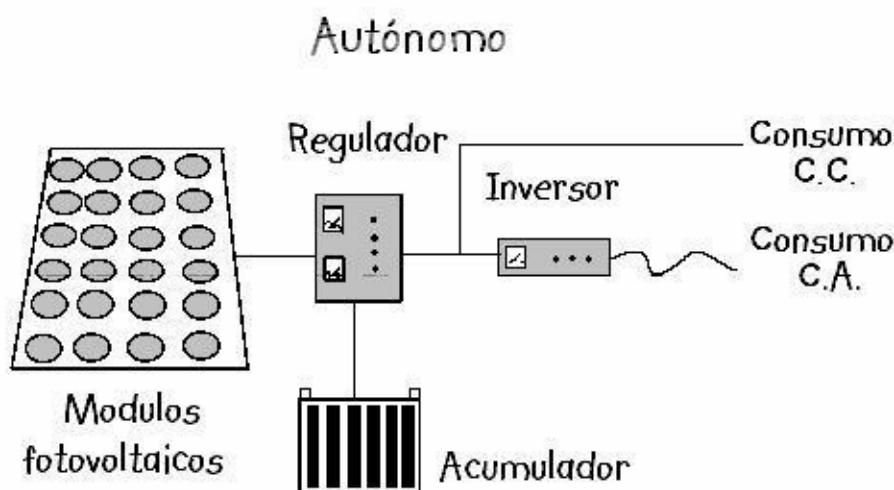


Figura. 1.2. Componentes en una instalación fotovoltaica autónoma

En las instalaciones fotovoltaicas autónomas (Fig 1.2) la función del regulador de carga es proteger los acumuladores contra la sobrecarga y sobre descarga. En caso de

sobrecarga, pone las placas en cortocircuito y corta la corriente hacia los acumuladores, o avisa al consumidor con una alarma, en el segundo caso, el de descarga excesiva, avisa con la alarma o corta el suministro cuando la cantidad de energía eléctrica del acumulador se pone por debajo de un nivel de seguridad.

Un buen sistema regulador no sólo permite aprovechar al máximo la energía sino que además protege las baterías y alarga su vida. El otro elemento importante de una instalación fotovoltaica es el grupo convertidor/inversor. Muchas de las instalaciones fotovoltaicas autónomas combinan consumos de corriente continua y corriente alterna. En algunos casos la tensión de los elementos de consumo de corriente continuo no coincide con la tensión proporcionada por el acumulador de la instalación, el cual requiere disponer de un convertidor de tensión. El inversor por su parte transforma la corriente continua, (12 o 24V) generada por la instalación en corriente alterna sinusoidal de 220 V a 50 Hz. ó 110 V a 60 Hz que es la corriente que utilizan la mayoría de aparatos.

Los sistemas conectados a la red eléctrica (Fig. 1.3) no tienen ni reguladores ni acumuladores. Se utilizan onduladores de corriente de mayor potencia que incluyen controles de fase para adecuar la corriente alterna producida para la red de consumo.

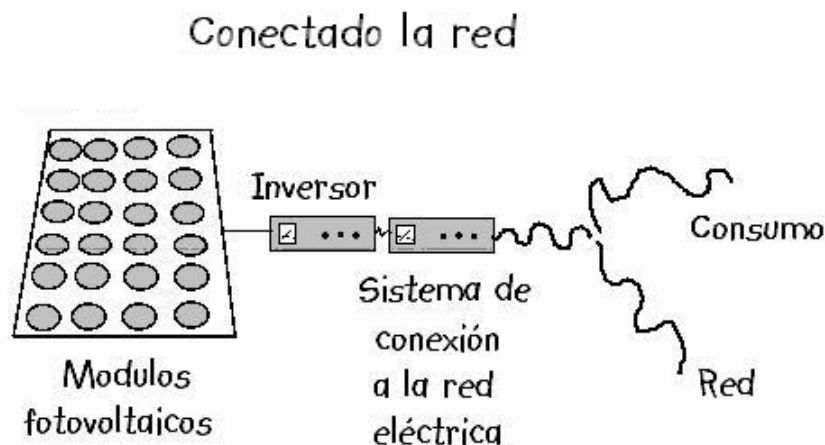


Figura. 1.3. Componentes en una instalación fotovoltaica conectada a la red

Los sistemas fototérmicos necesitan algunos componentes para la recogida directa de la energía solar, el elemento principal es el dispositivo artificial llamado colector solar, diseñado para recoger energía, a veces después de concentrar los rayos del sol. En los

procesos térmicos, la energía solar se utiliza para calentar un gas o un líquido que luego se almacena o se distribuye. Los colectores solares pueden ser de dos tipos principales: los de placa plana y los de concentración.

1.2.4. Aplicaciones

Una aplicación importante es la referente a los sistemas de aprovechamiento térmico. El calor recogido en los colectores puede destinarse a satisfacer numerosas necesidades. Por ejemplo, se puede obtener agua caliente para consumo doméstico o industrial, o bien para dar calefacción a nuestros hogares, hoteles, colegios, fábricas, etc. Incluso se pueden climatizar las piscinas y permitir el uso de agua caliente para el baño durante gran parte del año.

También, y aunque pueda parecer extraño, otra de las más prometedoras aplicaciones del calor solar será la refrigeración durante las épocas cálidas, precisamente cuando más soleamiento hay. En efecto, para obtener frío hace falta disponer de un "foco cálido", el cual puede perfectamente tener su origen en unos colectores solares instalados sobre el techo.

Las aplicaciones agrícolas son muy amplias. Con invernaderos solares pueden obtenerse mayores y más tempranas cosechas; los secaderos agrícolas consumen mucha menos energía si se combinan con un sistema solar, y, por citar otro ejemplo, pueden funcionar plantas de purificación o desalinización de aguas sin consumir ningún tipo de combustible. Las "células solares", dispuestas en paneles solares, ya producían electricidad en los primeros satélites espaciales. Actualmente se perfilan como la solución definitiva al problema de la electrificación rural, con clara ventaja sobre otras alternativas, pues, al carecer los paneles de partes móviles, resultan totalmente inalterables al paso del tiempo, no contaminan ni producen ningún ruido en absoluto, no consumen combustible y no necesitan mantenimiento. Además, y aunque con menos rendimiento, funcionan también en días nublados, puesto que captan la luz que se filtra a través de las nubes.

Los hornos solares son una aplicación importante de los concentradores de alta temperatura. El mayor, situado en Odeillo, en la parte francesa de los Pirineos, tiene 9 600 reflectores con una superficie total de unos 1 900 m² para producir temperaturas de hasta

4 000 °C. Estos hornos son ideales para investigaciones, por ejemplo, en la investigación de materiales, que requieren temperaturas altas en entornos libres de contaminantes.

Las células solares hechas con obleas finas de silicio, arseniuro de galio u otro material semiconductor en estado cristalino, convierten la radiación en electricidad de forma directa. Ahora se dispone de células con eficiencias de conversión superiores al 30%. Por medio de la conexión de muchas de estas células en módulos, el coste de la electricidad fotovoltaica se ha reducido mucho. El uso actual de las células solares se limita a dispositivos de baja potencia, remotos y sin mantenimiento, como boyas y equipamiento de naves espaciales.

1.2.5. Tendencias a Futuro

Cuando se empezaron a utilizar las primeras células fotovoltaicas en la industria aeroespacial en la década de los sesenta, nadie se imaginaba que treinta años después se utilizarían para suministrar electricidad para casas aisladas o para hacer funcionar teléfonos en autopistas. Esto ha sido posible gracias a la evolución de la tecnología que ha hecho que el precio de las células haya caído notablemente.

Consecuentemente, el mercado fotovoltaico a escala mundial ha experimentado un crecimiento espectacular: en 1 992 la producción de placas solares fotovoltaicas llegó a los 58MW duplicándose en sólo 7 años, para final de siglo se llegó a los 250MW/año.

Actualmente, uno de los campos en los que se está trabajando más intensamente es el desarrollo de células fotovoltaicas que tengan una eficiencia máxima, buscando nuevos materiales que rindan más.

La generación centralizada de electricidad a partir de energía solar está en desarrollo. Es necesario concentrar la radiación solar para que se puedan alcanzar temperaturas elevadas, de 300 °C hasta 1 000 °C, y obtener así un rendimiento aceptable en el ciclo termodinámico, que no se podría obtener con temperaturas más bajas. La captación y concentración de los rayos solares se hacen por medio de espejos con orientación automática que apuntan a una torre central donde se calienta el fluido, o con mecanismos más pequeños de geometría parabólica. El conjunto de la superficie reflectante y su

dispositivo de orientación se denomina "helióstato". De esta manera se podría tener una central térmica solar o termo-solar para generar electricidad como en una central térmica clásica, pero con mayor capacidad.

Un proyecto futurista propuesto para producir energía a gran escala propone situar módulos solares en órbita alrededor de la Tierra. En ellos la energía concentrada de la luz solar se convertiría en microondas que se emitirían hacia antenas terrestres para su conversión en energía eléctrica. Para producir tanta potencia como cinco plantas grandes de energía nuclear (de mil millones de vatios cada una), tendrían que ser ensamblados en órbita varios kilómetros cuadrados de colectores, con un peso de más de 4 000 toneladas; se necesitaría una antena en tierra de 8 m. de diámetro. Se podrían construir sistemas más pequeños para islas remotas, pero la economía de escala supone ventajas para un único sistema de gran capacidad.

CAPITULO II

SISTEMA DE SENSORES

2.1 INTRODUCCIÓN

La medición de magnitudes mecánicas, térmicas, eléctricas y químicas se realiza empleando dispositivos denominados sensores y transductores. El sensor es sensible a los cambios de la magnitud a medir, como una temperatura, una presión, o una concentración química. El transductor convierte estas mediciones en señales eléctricas, que pueden alimentar a instrumentos de lectura, registro o control de las magnitudes medidas. Los sensores y transductores pueden funcionar en ubicaciones alejadas del observador, así como en entornos inadecuados o impracticables para los seres humanos.

Algunos dispositivos actúan de forma simultánea como sensor y transductor. Por ejemplo un termopar consta de dos uniones de diferentes metales que generan una pequeña tensión que depende del diferencial térmico entre las uniones. El termistor es un reóstato especial, cuya resistencia varía según la temperatura. Para medir distancias se emplean condensadores de diseño especial, y para detectar la luz se utilizan fotocélulas. Para medir velocidades, aceleraciones o flujos de líquidos se recurre a otro tipo de dispositivos. En la mayoría de los casos, la señal eléctrica es débil y debe ser amplificada por un circuito electrónico.

En general, los transductores transforman un tipo de variable física (fuerza, presión, temperatura, velocidad, caudal, etc.) en otro (voltaje, corriente, resistencia, frecuencia, pulsos). El transductor necesita estar calibrado para ser útil como dispositivo de medida. La calibración es el procedimiento mediante el cual se establece la relación o función de transferencia entre la variable medida y la señal de salida convertida. Los transductores pueden clasificarse en dos tipos básicos dependiendo de la forma de la señal convertida.

Transductores Analógicos

Proporcionan una señal analógica continua, la cual puede ser voltaje, corriente o resistencia eléctrica. Esta señal puede ser tomada como un equivalente del valor de la variable física que se mide.

Transductores Digitales

Producen una señal de salida digital de dos estados que se transmite de modo serial a un instrumento de lectura o registro. Dependiendo de la interpretación que se le dé a esta salida digital, los transductores digitales pueden clasificarse en tres tipos:

Frecuencia. La salida binaria de este tipo de sensor se interpreta como una frecuencia, es decir, la cantidad de veces que la señal cambia en un segundo. El ejemplo más común de esta implementación, es el sensor de velocidad de viento el cual consiste en un rotor balanceado que gira a mayor o menor velocidad de acuerdo con la velocidad del viento, cada giro provoca el cierre de un interruptor que a su vez produce un pulso.

Pulsos. Es similar al anterior, pero a diferencia de este, nos interesa el valor acumulado, es decir, la cantidad total de eventos que se producen. Un ejemplo de este tipo de transductor es el pluviómetro; ya que como sabemos la lluvia es un parámetro que se presenta de manera imprevista y nos interesa registrar su valor acumulado.

Seriales. Este tipo de transductor produce a su salida una cadena de bits en un formato predeterminado utilizando diversos protocolos de comunicación (I²C, SDI₁₂, etc.). Existen en el mercado algunas versiones de estos transductores, un ejemplo es el termómetro digital DS1620. En este punto conviene señalar que los términos sensor y transductor, se usan indistintamente a lo largo del desarrollo del presente texto.

2.2 IDENTIFICACIÓN DE VARIABLES A MEDIR

Para tener en claro las variables físicas que se involucran dentro de la energía eólica y solar, primero se reconocerán todas las variables que son posibles de medir.

Dentro de la energía eólica tenemos diversas variables físicas estas son: velocidad, dirección, humedad, densidad, entre las principales.

Por otro lado, la energía solar también tiene diversas variables que son posibles medir: radiación directa, difusa y reflejada, radiación ultravioleta, radiación infrarroja, brillo solar, etc.

El “Sistema de monitoreo de energía eólica y solar con interfaz USB” se encarga específicamente de medir las siguientes variables físicas: Velocidad del viento y Radiación solar

2.2.1. Velocidad del Viento

El viento es la variable de estado de movimiento del aire. En meteorología se estudia el viento como aire en movimiento tanto horizontal como verticalmente. Los movimientos verticales del aire caracterizan los fenómenos atmosféricos locales, como la formación de nubes de tormenta.

La velocidad del viento se mide preferentemente en náutica en nudos (kt) y mediante la escala Beaufort. Esta escala comprende 12 grados de intensidad creciente que describen el viento a partir del estado de la mar. Esta descripción es inexacta pues varía en función del tipo de aguas donde se manifiesta el viento. Con la llegada de los modernos anemómetros, a cada grado de la escala se le ha asignado una banda de velocidades medidas por lo menos durante 10 minutos a 10 metros de altura sobre el nivel del mar.

Actualmente, la escala Beaufort tiende a sustituirse por las mediciones precisas en m/s.

$$1 \text{ kt} = 1.8 \text{ km/h} \text{ ó } 1 \text{ kt} = 0.5 \text{ m/s.}$$

2.2.2. Radiación Solar

Energía radiante producida en el Sol como resultado de reacciones nucleares de fusión. Llega a la Tierra a través del espacio en cuantos de energía llamados fotones, que interactúan con la atmósfera y la superficie terrestres. La intensidad de la radiación solar en el borde exterior de la atmósfera, si se considera que la Tierra está a su distancia promedio del Sol, se llama constante solar, y su valor medio es $1,37 \times 10^6 \text{ erg/s/cm}^2$, o unas 2 cal/min/cm^2 . Sin embargo, esta cantidad no es constante, ya que parece ser que varía un 0,2% en un periodo de 30 años. La intensidad de energía real disponible en la superficie terrestre es menor que la constante solar debido a la absorción y a la dispersión de la radiación que origina la interacción de los fotones con la atmósfera.⁽²⁾

La energía calorífica de la radiación solar es la generatriz de todos los procesos meteorológicos y climáticos que se dan en la tierra. Al incidir sobre el planeta, atraviesa el gas atmosférico sin apenas calentarlo; en cambio sí calienta a la superficie terrestre que es la que acaba transmitiendo el calor al aire atmosférico en contacto con ella. Así pues, es la Tierra la que calienta directamente la atmósfera y no la radiación solar. Esto tiene una importante trascendencia para entender la dinámica de todos los procesos que se dan en meteorología.



Figura. 2.1. La tierra del ecuador se calienta más por la acción solar que la de los polos, debido a que recibe más cantidad de radiación por unidad de superficie

Sin embargo, no toda la superficie de la Tierra recibe por igual la misma energía: los polos son las que menos y las zonas ecuatoriales son las que más como se observa en la

(2) SENAMHI. GUÍA BÁSICA DE METEOROLOGÍA GENERAL. 2004.

Figura 2.1. De este modo, la superficie de la Tierra no transmite de una forma uniforme el calor al aire que tiene sobre ella. Esto origina que se produzcan intercambios térmicos entre las zonas más calientes y las más frías para restablecer el equilibrio: el aire caliente se desplaza hacia los polos y el aire frío hacia el ecuador. De este modo, las masas de aire nivelan y suavizan el clima en la Tierra y establecen los principios de la circulación general.

Estas variables son las más importantes para lo que es generación de energía, ya sea eólica o solar. Es por esto que se escogieron esas variables físicas.

2.3 SISTEMA EÓLICO

2.3.1. Selección del Sensor

El sensor que se escogió para medir la velocidad del viento es el Transmisor de Viento – Compacto de la marca Thies Clima modelo 4.3519.00.000. Puesto que este sensor trabaja a rangos de voltaje compatibles con TTL y porque su salida es una señal digital.

2.3.1.1. Transmisor de Viento – Compacto 4.3519.00.000

Este sensor de viento mide y trasmite la velocidad horizontal de viento. Los valores a la salida del sensor están disponibles como señal digital. Lo cual facilita la conexión con equipos como dataloggers, displays, tecnología TTL, etc.

Este instrumento puede ser usado en un rango de velocidad de viento desde 0,5 – 50 m/s. Trabaja en un rango de temperatura de -30 °C a +70°C. El voltaje de alimentación Vcc puede ser de 4 – 18 V DC y la Salida es un pulso cuya amplitud es Vcc.

Debido a que la salida es una señal digital no es necesario ningún circuito de acondicionamiento.

2.3.1.2. Función de Transferencia

Primero es necesario encontrar la pendiente de la función de transferencia. Para ello utilizamos los extremos de frecuencia y velocidad como sigue (Para mayor detalle referirse al ANEXO A – Hoja de datos técnica del sensor):

$$m = \frac{F_2 - F_1}{V_2 - V_1}$$

$$m = \frac{630 \text{ Hz} - 2 \text{ Hz}}{50 \text{ m/s} - 0,5 \text{ m/s}}$$

$$m = 12,68$$

$$V [m/s] = \frac{F [\text{Hz}]}{m} + b$$

$$b = V - \frac{F}{m}$$

$$b = 50 \text{ m/s} - \frac{630 \text{ Hz}}{12.68}$$

$$b = 0.32 \left[\frac{\text{Hz} \cdot \text{m}}{\text{s}} \right]$$

$$V [m/s] = 0.07881 \cdot F [\text{Hz}] + 0.32$$

2.4 SISTEMA SOLAR

2.4.1. Selección del Sensor

Para motivo del presente proyecto y por facilidad de adquisición se utilizará el Sensor de Radiación Solar SSR 81.

2.4.1.1. Standar Silizium Radiometer SSR 81

El sensor de radiación utilizado es de la marca Tricon, modelo SSR 81, este es un sensor de bajo costo, que sirve para medir la radiación solar en W/m^2 . Similar a un piranómetro (instrumento de medición de la energía solar global), tiene un cristal de cuarzo

que encapsula una celda de silicio monocristalino que provee un voltaje proporcional a la fuerza de radiación solar. Esta celda solar genera niveles de voltaje dependiendo del valor de radiación global recibido. Su sensibilidad es de 74.5 mV por cada 1 000 W/m², este dato es proporcionado por el fabricante.

2.4.1.2. Circuito de Acondicionamiento

El circuito de acondicionamiento utiliza dos operacionales LM324. El primero sirve para acoplar impedancias, es decir esta configurado como seguidor de voltaje; el segundo operacional esta configurado como amplificador con una ganancia aproximada de 50, esto con el fin de obtener un valor de voltaje dentro del intervalo de interés (0 y 5 voltios). A continuación en la figura 2.2, podemos observar el circuito implementado.

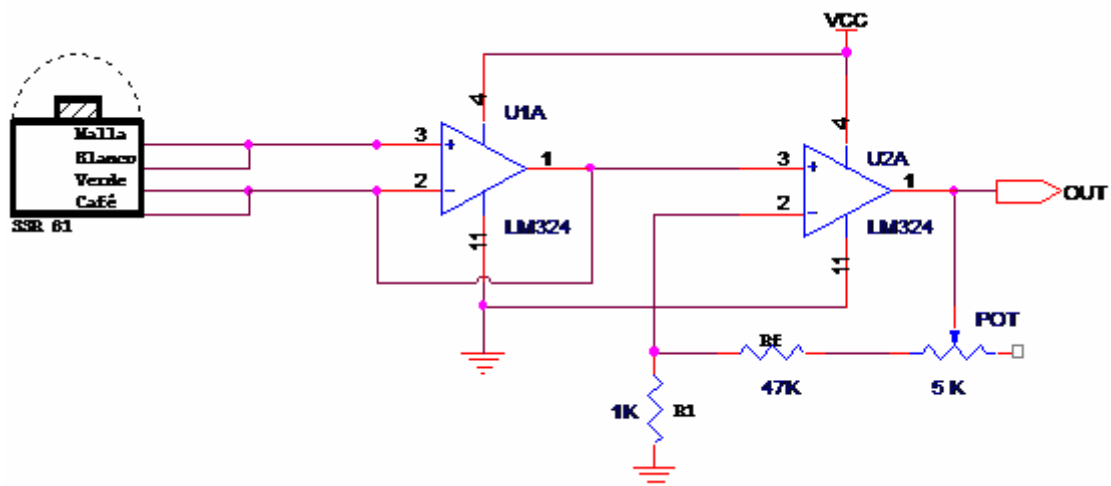


Figura. 2.2. Circuito de acondicionamiento del sensor de radiación solar

Con $R_F = 47 \text{ k}\Omega$, $R_1 = 1 \text{ k}\Omega$ y el potenciómetro de $5\text{K}\Omega$, se obtiene una ganancia real de 50. Utilizando esta ganancia, podemos obtener el valor de voltaje que corresponden a $1\ 000 \text{ W/m}^2$ utilizando la sensibilidad.

$$V_{out} = A * V_{in}$$

$$V_{out} = 50 * 74.5\text{mV}$$

$$V_{out} = 3.725 \text{ V}$$

Con esto podemos decir que para una radiación global de 1000 W/m^2 , el voltaje obtenido es de 3.725 V y para un nivel de radiación nula, es decir cero; el nivel de voltaje es cero.

2.4.1.3. Función de Transferencia

Primero es necesario encontrar la pendiente y el offset de la función de transferencia. Para ello utilizamos los extremos de radiación y voltaje como sigue (Para mayor detalle referirse al ANEXO B – Hoja de datos técnica del sensor):

$$m = \frac{R_2 - R_1}{V_2 - V_1}$$

$$m = \frac{1000 \text{ W / m}^2 - 0 \text{ W / m}^2}{3.725 \text{ V} - 0 \text{ V}}$$

$$m = 268.45$$

$$b = R - m * V$$

$$b = 1000 - 268.45 * 3.725$$

$$b = 0$$

$$R = 268.45 * V$$

2.5 MICROCONTROLADOR 16F877

Considerando el momento actual y comparando los parámetros fundamentales con los modelos comerciales de otros fabricantes y las aplicaciones más habituales a las que se destinan los microcontroladores, opinamos que casi en un 90 % de los casos la elección de una versión adecuada de PIC es la mejor solución. Sin embargo, otras familias de microcontroladores son más eficaces en aplicaciones concretas, especialmente si predomina una característica especial.

Pero no queremos dejar pasar la ocasión de afirmar que en la actualidad los PIC tienen "algo" que les agrada a los diseñadores. Pueden ser la velocidad, el precio, la facilidad de uso, la información, las herramientas de apoyo. Quizás un poco de todo es lo

que produce esa imagen de sencillez y utilidad. Es muy posible que mañana otra familia de microcontroladores le arrebatase ese "algo". Es la ley del mercado y la competencia.

Generalmente para la mayoría de los sistemas de control se necesita conocer la capacidad y el tipo de las memorias, el número de líneas de entrada y salida, temporizadores, conversores análogo digital, así como circuitos auxiliares, que son parámetros que dependen exclusivamente de la aplicación y varían mucho de unas situaciones a otras. Quizás se pueda considerar la decisión más importante del proyecto la elección del modelo de microcontrolador. Para el presente proyecto hemos decidido utilizar el PIC 16F877 que para esta aplicación es el más apropiado, por lo que estamos familiarizados con su utilización, posee un conversor A/D interno de 10 bits y tiene el número necesario de puertos. Además sus elementos de programación, ensamblaje y grabación son genéricos y de fácil adquisición.

El PIC 16F877 pertenece a una subfamilia de microcontroladores de la gama media. Es apropiado para aplicaciones de control de máquinas programables, dispositivos y sensores. Las características y las funciones de interés de este PIC y por las cuales se lo escogió son las que se indican a continuación.

2.5.1. Características Generales

- Procesador de arquitectura RISC avanzada.
- Juego de 35 instrucciones con 14 bits de longitud.
- Frecuencia de 20 MHz.
- Capacidad de memoria: Dispone de 8Kx14 bits de memoria FLASH mejorada, 256 bytes de memoria de datos EEPROM y hasta 368 bytes de Memoria de Datos RAM.
- Hasta 14 fuentes de interrupción internas y externas.
- Perro Guardián (WDT).
- Código de protección programable.
- Modo SLEEP de bajo consumo.
- Programación serie en circuito con dos pines.
- Voltaje de alimentación comprendido entre 2 y 5.5 V.

- Bajo consumo (menos de 2 mA a 5V y 5 MHz).

2.5.2. Dispositivos Periféricos

- Dispone de 5 puertos de E/S con un total de 33 líneas para conectar a los periféricos exteriores con una puerta paralela esclava.
- Proporciona de 5 a 8 canales de convertidores A/D de 10 bits.
- Posee Temporizadores-Contadores de 8 bits y 16 bits.
- Dos módulos de Captura-Comparación-PWM.
- Conversor A/D de 10 bits.
- Puerto Serie Síncrono (SSP) con SPI e I²C.
- USART.
- Puerta Paralela Esclava (PSP).

2.5.3. Pines de Propósito General

- OSC1/CLKIN: Entrada de cristal de cuarzo o del oscilador externo.
- OSC2/CLKOUT: Salida del cristal de cuarzo. En modo RC el pin OSC2 saca la cuarta parte de la frecuencia que se introduce por OSC1, que determina el ciclo de instrucción.
- VSS: Conexión a tierra.
- VDD: Entrada de la alimentación positiva.
- MCLR/VPP/THV: Entrada de RESET o entrada del voltaje de programación o voltaje alto en el modo test.

2.5.4. Puertos de Entrada Salida (E/S)

- Posee cinco puertos, todas las líneas de estos puertos son multifuncionales, es decir, realizan diferentes funciones según como estén programadas. Sin embargo, todas ellas tienen la capacidad de trabajar como líneas de E/S digitales.

2.5.4.1. Puerto A

- RA0/AN0: Puede actuar como línea digital de E/S o como entrada analógica al convertor A/D (canal 0).
- RA1/AN1: Igual que la RA0/AN0.
- RA2/AN2/VREF-: Puede ser línea digital de E/S, entrada analógica o entrada del voltaje negativo de referencia.
- RA3/AN3/VREF+: Línea digital de E/S, entrada analógica o entrada del voltaje de referencia positivo.
- RA4/TOCKI: Línea digital de E/S o entrada del reloj del Timer0. Salida con colector abierto.
- RA5/SS/AN4: Línea digital de E/S, entrada analógica o selección como esclavo del puerto serie síncrono.

2.5.4.2. Puerto B

- RB0/INT: Línea digital de E/S o entrada de petición de interrupción externa.
- RB1: Línea de E/S digital.
- RB2: Línea de E/S digital.
- RB3/PGM: Línea digital de E/S o entrada del voltaje bajo para programación.
- RB4: Línea de E/S digital.
- RB5: Línea de E/S digital.
- RB6/PGC: Línea digital de E/S. En la programación serie recibe las señales de reloj.
- RB7/PGD: Línea digital de E/S. En la programación serie actúa como entrada de datos.

2.5.4.3. Puerto C

- RCO/T10SO/T1CKI: Línea digital de E/S o salida del oscilador del Timer1 o como entrada de reloj Timer1.
- RC1/T1OSI/CCP2: Línea digital de E/S o entrada al oscilador del Timer1 o entrada al modulo Captura2/salida Comparacion2/salida de PWM2.

- RC2/CCP1: E/S digital. También puede actuar como entrada Captural/Salida Comparación/salida de PWM1.
- RC3/SCK/SCL: E/S digital o entrada de reloj serie síncrona/salida de los modos SPI e I²C.
- RC4/SDI/SDA: E/S digital o entrada de datos en modo SPI o I/O datos en modo I²C.
- RC5/SDO: E/S digital o salida de datos en modo SPI.
- RC6/TX/CK: E/S digital o pin del transmisor del USART asíncrono o como reloj síncrono.
- RC7/RX/DT: E/S digital o receptor del USART asíncrono o como datos en el síncrono.

2.5.4.4. Puerto D

- RDO/PSPO-RD7/PSP7: Los 8 pines de este puerto pueden actuar como líneas de E/S digitales o como líneas para la transferencia de información en la comunicación del puerto paralelo esclavo. Solo están disponibles en los PIC16F877.

2.5.4.5. Puerto E

- REO/RD/AN5: E/S digital o serial de lectura para el puerto paralelo esclavo o entrada analógica (canal 5).
- RE1/WR/AN6: E/S digital o señal de escritura en el puerto paralelo esclavo o entrada analógica al convertor A/D (canal 6).
- RE2/CS/AN7: E/S digital o activación/desactivación del puerto paralelo esclavo o entrada analógica (canal 7).

2.5.4.6. Conversor A/D

El convertor A/D es posiblemente el recurso del PIC 16F877 que más aplicaciones tiene. La tarea de conversión es siempre igual para cualquier sensor utilizado. Posee un convertor A/D de 10 bits de resolución y 8 canales de entrada.

La máxima lectura del convertidor análogo a digital se obtiene cuando la entrada análoga sea 5 voltios. La resolución que tiene cada bit procedente de la conversión tiene un valor que es función de la tensión de referencia V_{REF} , de acuerdo con la fórmula siguiente:

$$resolución = \frac{V_{REF+} + V_{REF-}}{1024} = \frac{V_{REF}}{1024}$$

La tensión de referencia puede implementarse con la tensión interna de alimentación V_{DD} , o bien, con una externa que se introduce por el pin RA3/AN3/ V_{REF+} , en cuyo caso la polaridad negativa se aplica por el pin RA2/AN2/ V_{REF-} .

Para un voltaje de referencia $V_{REF+} = 5 V_{DC}$ y $V_{REF-} = 0$, la resolución del conversor A/D es de 4,8 mV/bit. La tensión de referencia determina los límites máximo y mínimo de la tensión analógica que se puede convertir.

El conversor A/D es el único dispositivo que puede funcionar en modo Reposo (SLEEP), para ello el reloj del conversor deberá conectarse al oscilador RC interno. Si se elige este modo, la conversión puede seguir funcionando, en los restantes modos se aborta la conversión y se desactiva el conversor A/D.

El valor de TAD se selecciona por software, y es el tiempo que dura la conversión de cada bit y en el caso de trabajar con valores digitales de 10 bits en el PIC16F877 nunca debe ser menor de 1,6 us. Se requiere un tiempo mínimo de 12 TAD.

Es importante tomar en cuenta que los 16 bits que forman la concatenación unas veces tiene a 0 los 6 bits más significativos y otras los 6 bits menos significativos (alineación a la derecha o izquierda), se debe verificar este aspecto cuando se realice la programación.

Los pasos que se deben seguir para realizar una conversión A/D son:

- Configurar el modulo A/D

- Configurar los pines que actuarán como entradas analógicas, las que trabajan como E/S digitales y las utilizadas para la tensión de referencia.
- Seleccionar el reloj de la conversión.
- Seleccionar el canal de entrada A/D.
- Activar el modulo A/D.
- Tiempo de espera para que transcurra el tiempo de adquisición.
- Inicio de la conversión.
- Tiempo de espera para completar la conversión A/D que puede detectarse.
- Leer el resultado en los 10 bits válidos.

2.6 ESTUDIO DE MEMORIA

El sistema de memorias es el encargado de almacenar los datos digitalizados convenientemente hasta que exista vía libre para poder transmitirlos. Para realizar este cometido utilizaremos memorias I²C CMOS Serial EEPROM, el motivo es que el espacio de memoria EEPROM del PIC 16F877 es de 256 x 8 bytes, espacio que resulta insuficiente para el propósito del circuito.

Las memorias I²C presentan muchas ventajas con respecto a las memorias paralelas de uso común como son:

- Los datos pueden leerse y escribirse usando solamente dos pines.
- Es posible conectar hasta 8 memorias para obtener grandes capacidades.
- El circuito para conectar varias memorias seriales es relativamente sencillo.
- Realizan la comunicación mediante protocolo I²C el cual es muy usado para comunicación entre dispositivos electrónicos, y que además es soportado por el microprocesador.
- Usan el mismo protocolo (I²C) que el Reloj de Tiempo Real (RTC), razón por la cual se pueden manejar con los mismos dos pines que controlan el RTC.
- Utilizan muy poco espacio en la placa base.
- Y al igual que las memorias EEPROM paralelas, retienen la información aún sin energía.

En contraste presentan algunas desventajas como son su baja velocidad de transferencia, ya que los datos se envían bit a bit, y sus bajas capacidades de almacenamiento. No obstante estas desventajas no constituyen mayor problema para el diseño ya que no se requieren altas velocidades ni se manejarán grandes volúmenes de información.

El banco de memorias estará constituido por 2 memorias de 256 kbits (32 kbytes), una para cada sensor, esto significa 32K x 8 (256Kbit), es decir 32.000 bytes (datos), para cada sensor. Para el caso del presente circuito cada dato se almacenará en 1 byte lo que significa 32.000 datos por memoria; lo cual satisface las necesidades del presente estudio.

Estas memorias trabajan a corrientes muy bajas, máximo 3 mA y a un rango de voltaje de 2.5 a 5.5 V por lo que es compatible con tecnología TTL. Cada dispositivo cuenta con 8 pines que tienen diferente funcionalidad (Fig. 2.3).

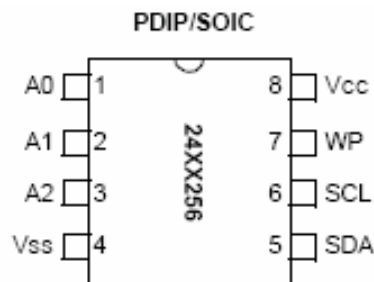


Figura. 2.3. Distribución de pines memoria I²C

Tabla. 2.1. Distribución de pines memoria I²C

A0, A1 y A2:	Pines para las direcciones
Vss	Tierra
Vcc	Voltaje de alimentación (+2.5 V a +5.5 V)
WP	Protección de escritura
SCL	Reloj serial
SDA	Datos (bi-direccional)

Los pines del 1 al 3 (A0, A1, A2) son los pines de dirección de cada memoria, en conjunto son el código de tres bits que permite conectar hasta ocho dispositivos del mismo tipo. Por ejemplo si se trata de la primera memoria los tres pines deberán estar conectados a GND, si se trata de la segunda memoria A0 deberá estar a uno, mientras A1 y A2 deben conectarse a cero, y así sucesivamente. Lo dicho anteriormente puede verse mejor en la siguiente tabla.

Tabla. 2.2. Pines de dirección memoria I²C

Dispositivo Seleccionado	A2	A1	A0
Memoria serial #1	0	0	0
Memoria serial #2	0	0	1
Memoria serial #3	0	1	0
Memoria serial #4	0	1	1
Memoria serial #5	1	0	0
Memoria serial #6	1	0	1
Memoria serial #7	1	1	0
Memoria serial #8	1	1	1

Utilizando estos pines se puede saber hacia qué dispositivo se está enviando los datos, para ello se usarán los bits de dirección contenidos en el byte de control de I²C que se detallará más adelante.

Los pines cinco y seis (SDA y SCL) son los pines datos y reloj respectivamente; a través de SDA podemos leer y escribir datos en la memoria, mientras que SCL es una entrada de reloj para que la memoria pueda leer o escribir los datos bit a bit. La señal de reloj debe ser proporcionada por el microprocesador que realiza los procesos de lectura y escritura. Si se acoplan mas de una memoria serial, todos los pines de datos se conectan entre si al igual que las entradas de reloj. Esto es lo que se conoce como un bus I²C. Cabe mencionar que estas dos entradas son de colector abierto, por lo tanto para su correcto desempeño es necesario colocar resistencias de PULL-UP.

El pin 7 (WP) es de protección contra escritura si se coloca a uno lógico solo es posible leer los datos mas no realizar escritura. No se aprovechará esta facilidad y se colocará esta entrada a cero. Los pines 4 y 8 son GND y V_{CC} , respectivamente.

En el ANEXO C, se hace un extracto de lo más sobresaliente de las hojas técnicas de la memoria 24LC256.

2.6.1. Reloj Calendario de Tiempo Real

Es un dispositivo que proporciona al sistema la fecha y hora exactas. Este circuito es necesario para saber a que hora y cuando se realizó una medición. De él podemos obtener una base de tiempo confiable; nos proporciona: año, mes, día, horas minutos y segundos.

Se utilizará el Circuito Integrado DS1307; este integrado al igual que las memorias, utiliza I^2C , lo cual facilita su conexión al circuito y a la vez permite su manejo a través del mismo bus I^2C que controla las memorias. Funciona mediante un oscilador de cristal de 32.768 Khz. e incorpora una batería de litio de 3 VDC para que el reloj no se desigale aún sin ser alimentado. El proceso de adquisición del tiempo es bastante simple, y se reduce a leer las localidades de memoria que contienen las variables temporales empleando I^2C . El mapa de memoria del RTC puede verse en la tabla a continuación.

Tabla. 2.3. Mapa de memoria del RTC

Localidad de memoria	Variables
00H	Segundos
01H	Minutos
02H	Horas
03H	# día de la semana
04H	# día del mes
05H	# de mes
06H	Año (2 últimos dígitos)

07H	Control
08H	56 localidades de memoria RAM para uso general
3FH	

Es importante mencionar que las variables temporales del RTC se encuentran en formato BCD. Y para iniciar el conteo debemos poner el bit más significativo de los segundos a cero, lo cual se realiza escribiendo cualquier valor de segundos de 0 a 60 en el registro.

Físicamente el dispositivo consta de 8 pines; los dos primeros X1 y X2 son las entradas para el oscilador de cristal de 32.768 Khz. al tercer pin (VBAT) se conecta la batería de respaldo de 3 V; el cuarto y octavo son GND y V_{CC} respectivamente; los pines cinco y seis (SDA y SCL) son los que se conectan al bus I²C y del mismo modo que en las memorias, son salidas de colector abierto; finalmente el pin siete (SQW/out) es una salida de reloj que puede ser configurada mediante el registro de control.

El registro de control mostrado a continuación, es muy importante para el funcionamiento del RTC.

Tabla. 2.4. Registro de control del RTC

OUT	0	0	SQWE	0	0	RS1	RS0
-----	---	---	------	---	---	-----	-----

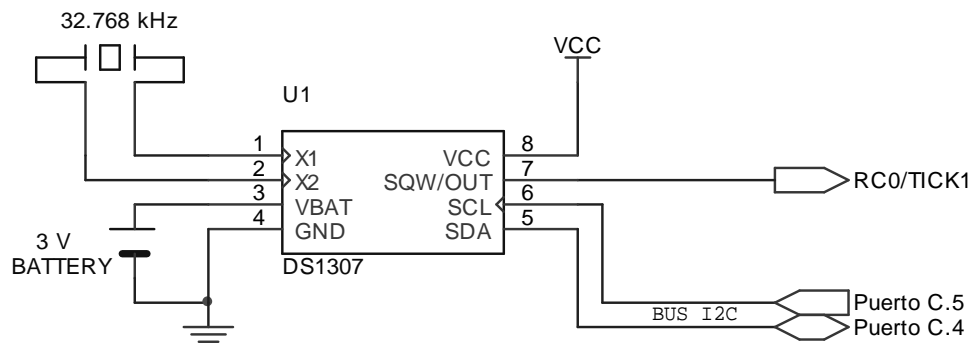
El bit más significativo (OUT) define el estado del pin siete del chip, siempre y cuando la salida de reloj a través de este pin esté deshabilitada. Para habilitar la salida de reloj es necesario colocar a 1 el bit 4 del registro de control (SQWE). La frecuencia de la señal de reloj queda determinada por los dos primeros bits de este mismo registro, de acuerdo a la siguiente tabla.

Tabla. 2.5. Frecuencia de la señal de reloj del RTC

RS1	RS0	FRECUENCIA DE SALIDA
0	0	1 hz.
0	1	4.096 Khz.
1	0	8.192 Khz.
1	1	32.768 Khz.

La salida de reloj configurada para producir 1 Hz es de gran importancia en el diseño, ya que esta frecuencia conectada al TIMER1 del microcontrolador, configurado como contador, constituye un eficiente contador de minutos. Este pequeño arreglo nos da la base de tiempo para realizar las mediciones comparando el contador con los tiempos de muestreo de cada canal.

El diagrama de conexiones de este integrado se observa en la Figura 2.4.

**Figura. 2.4. Diagrama de Conexiones del RTC**

En el ANEXO D, se presenta la hoja técnica con las especificaciones del fabricante, del RTC.

2.6.2. Modo de Almacenamiento de Datos

Para almacenar un dato binario resultante de una medición, se debe tener en cuenta que el valor binario por si mismo carece de significado, razón por la cual además de dicho

valor, es necesario almacenar información adicional que permita identificar la procedencia de dicho valor y el momento en el cual fue tomado. Por este motivo se almacenará el número de canal, fecha y hora a los cuales se efectuó la medición. De este modo por cada medición se guardarán en memoria ocho bytes en el orden que muestra la tabla.

Tabla. 2.6. Bytes de almacenamiento de canal, fecha y hora de la medición

# de Byte	Información Contendida
1° byte	MSB resultante del 1er. Sensor
2° byte	LSB resultante del 1er. Sensor
3° byte	MSB resultante del 2do. Sensor
4° byte	LSB resultante del 2do. Sensor
5° byte	Horas
6° byte	Minutos
7° byte	Día
8° byte	Mes

Con toda la información anterior, se está en condiciones de estimar el tiempo que pueden permanecer los datos en el banco de memorias sin que los nuevos datos se sobrescriban en los anteriores.

Las peores condiciones en cuanto a uso del banco de memorias, se suscitan cuando tenemos los dos canales activos y tomando datos al mínimo tiempo, que es cada cinco minutos, esto supone que habrán 12 mediciones cada hora por canal, en el día se tendrán $12 \times 24 = 288$ mediciones por canal. Considerando todos los canales, se habrán realizado $288 \times 2 = 576$ mediciones. Y como cada canal almacena 8 bytes por cada medición, tendremos un total de $576 \times 8 = 4608$ bytes/día. Si la capacidad total del banco de memorias es $32768 \times 2 = 65536$ bytes, tenemos:

$$65536 \text{ Bytes} * \frac{\text{día}}{4608 \text{ Bytes}} = 14.22 \text{ días}$$

Es decir, que los datos pueden permanecer sin descargarse dos semanas en la peor situación en cuanto a uso de la memoria.

2.7 TARJETA DE ALMACENAMIENTO DE DATOS

2.7.1. Código del Circuito de Almacenamiento de Datos

Una parte fundamental del proyecto es el desarrollo del programa para que el Hardware trabaje de acuerdo a lo planificado, por tal motivo la herramienta de programación que se utilice, debe proporcionar todas las facilidades para cumplir con los objetivos.

En este caso, se procedió a trabajar con la herramienta de programación de microcontroladores llamada MicroCode Studio Plus, el cual trabaja con las librerías de Pic Basic Pro. Esta es una herramienta poderosa desarrollada para simplificar el tiempo de programación y aprovechar todas las potencialidades que poseen los Microcontroladores de la gama media.

2.7.1.1. Introducción a MicroCode Studio

El Microcode Studio Plus es una poderosa herramienta de Desarrollo Visual Integrada, denominado (IDE) por sus siglas en Inglés, con capacidad de depuración en Línea, diseñado para trabajar con Micro Engineering Labs con los compiladores PIC Basic y Pic Basic Pro, a continuación en la Fig. 2.5 les presentamos el interfaz del programa MicroCode Studio Plus.

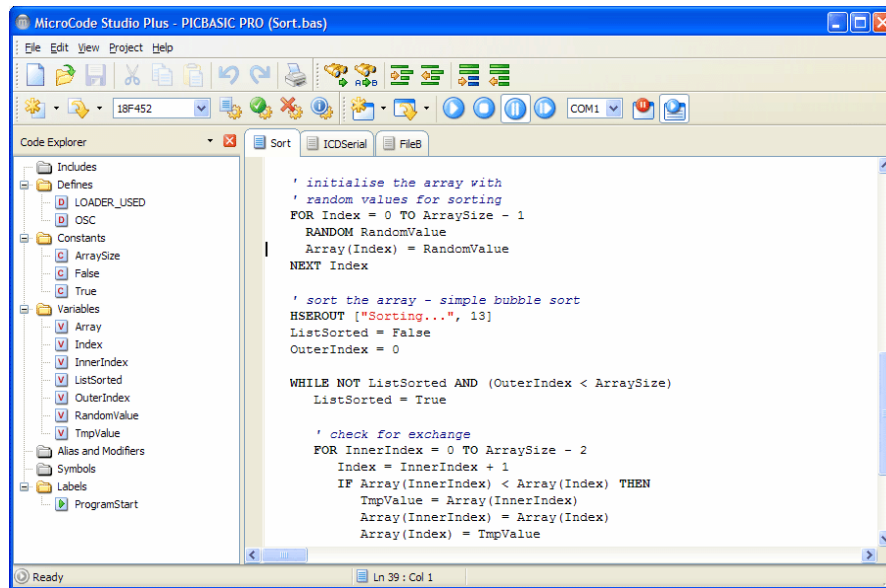


Figura. 2.5. MicroCode Studio Plus

Esta formidable herramienta de trabajo, posee algunas funciones muy interesantes como las siguientes:

Ventana de Comunicación Serial, la cual habilita al usuario a conectarse y monitorear el puerto serial. En la Fig. 2.6 que presentamos a continuación se puede visualizar el modo en que se presenta esta ventana.

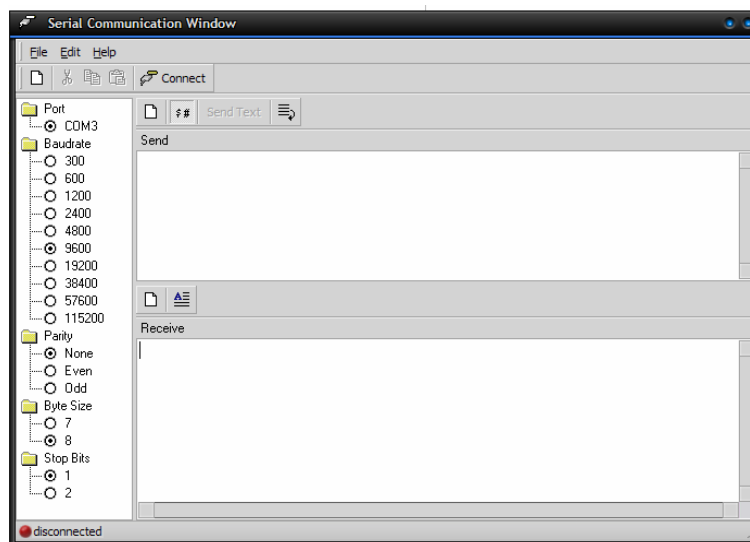


Figura. 2.6. Ventana de comunicación serial

Comandos de Referencia, que permiten desarrollar el programa de una manera mas comprensible, ya sea para el encendido de un LED con opciones como HIGH, asignándole un alto al puerto que deseamos que se encienda, hasta instrucciones para comunicación USB como el USBINIT, USBIN, USBOUT, que son importantes para el desarrollo del interfaz de comunicación de nuestro Proyecto.

Entre los comandos de referencia mas utilizados tenemos los siguientes:

USBINIT

Este soporte ha sido añadido para los dispositivos USB de baja velocidad 16C745 y 16C765. El comando USBINIT debe ser una de las primeras declaraciones en el programa. Este inicializará la parte de comunicación USB del PIC y espera hasta que el bus USB este configurado y habilitado. Este comando interactúa con los archivos propios de la microchip para ejecutar la comunicación. Para aplicaciones particulares, estos archivos deben ser modificados.

USBIN

La declaración del comando USBIN es la siguiente:

USBIN Endpoint, Buffer, CountVar, Label

USBIN consigue cualquier dato en el puerto USB para el Endpoint y lo coloca en el Buffer. El Buffer debe ser una cadena de bytes, de longitud conveniente para contener los datos. CountVar, contendrá el número de bytes transferidos al en Endpoint. El Label, es la etiqueta en la que el programa estará ejecutándose si no encuentra datos disponibles en el bus.

USBOUT

La declaración del comando USBOUT es la siguiente:

USBOUT Endpoint, Buffer, Count, Label

USBOUT cuenta el número de bytes del Buffer y los envía al USB Endpoint. Si el Buffer USB no tiene suficiente espacio para los datos debido a una transmisión pendiente, ningún dato será transferido y el programa seguirá ejecutándose en el Label.

SEROUT2

La declaración para el comando SEROUT2 es la siguiente

```
SEROUT2 DataPin{\FlowPin},Mode,{Pace,} {Timeout,Label,}[Item...]
```

SEROUT2 envía uno o varios Items a DataPin en formato estándar asíncrono. DataPin automáticamente es una salida de datos. FlowPin (característica opcional) está definido como entrada. DataPin y FlowPin puede ser constantes, 0 - 15, o una variable que contiene un número 0 - 15 (p.ej. B0) o pueden tener el nombre del puerto (PORTA.0). El pin de control de flujo opcional, FlowPin, puede ser incluido para impedir a datos invadir al receptor. Una Interrupción opcional y el Label, pueden ser incluidas para permitir al programa seguir si el FlowPin no se cambia a estado permitido dentro de una cierta cantidad de tiempo. La interrupción es especificada en las unidades de 1 milisegundo.

En algunos casos, la tasa de transmisión de instrucciones SEROUT2 puede presentar caracteres demasiado rápido al dispositivo de recepción.

I2CREAD e I2CWRITE

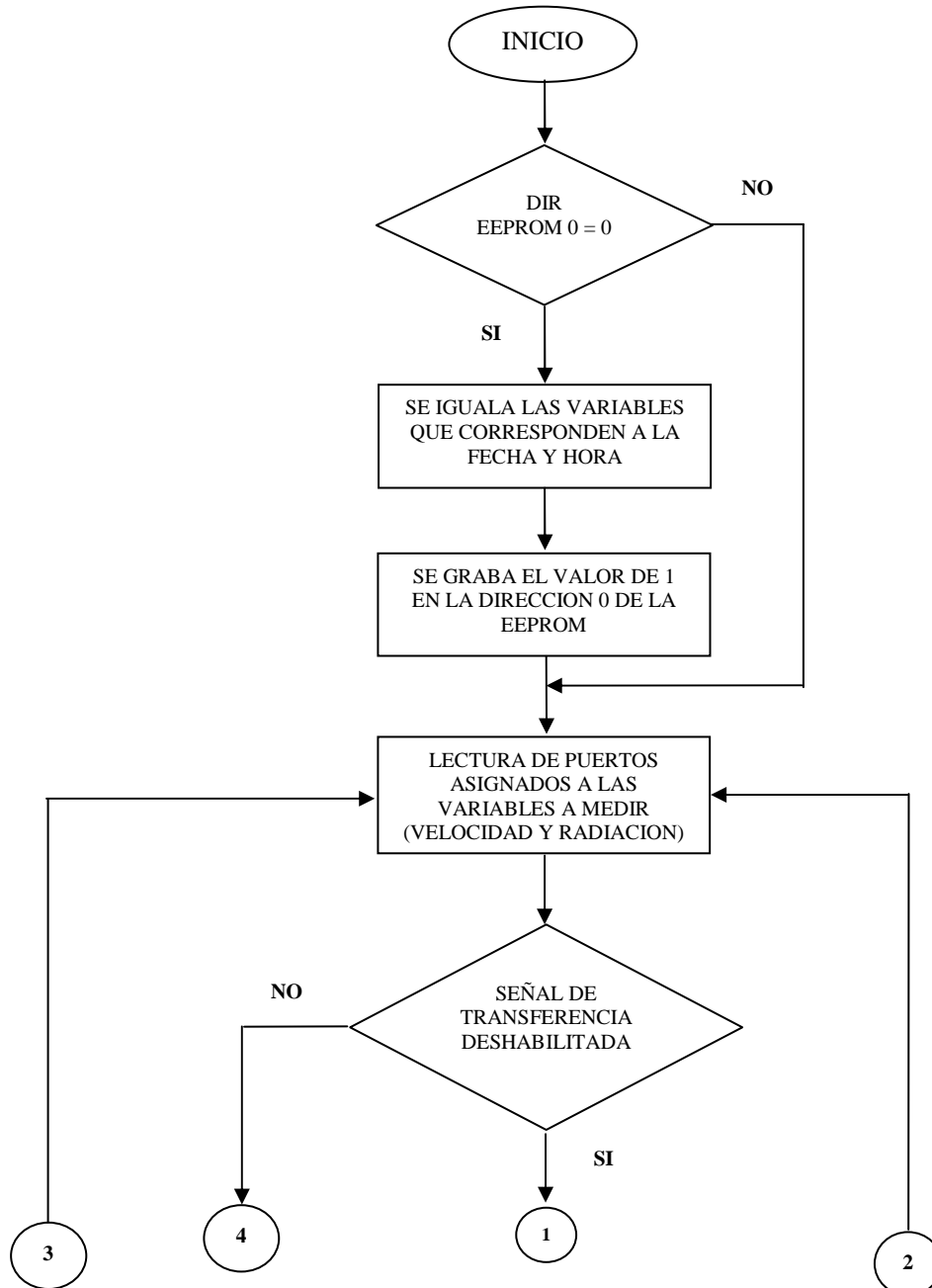
Las declaraciones de estos comandos son las siguientes:

```
I2CREAD DataPin,ClockPin,Control,{Address,}[Var{,Var...}]{,Label}
```

```
I2CWRITE DataPin,ClockPin,Control,{Address,}[Value{,Value...}]{,Label}
```

I2CREAD e I2CWRITE son comandos usados para leer y escribir datos en memorias EEPROM seriales conectadas con dos hilos (el DataPin y el ClockPin), que forman el bus I²C, lo que permite guardar datos en una memoria externa en forma permanente, de modo que puedan permanecer almacenados aún cuando se suspenda la energía en el circuito.

2.7.1.2. Diagrama de Flujo del Código del Programa



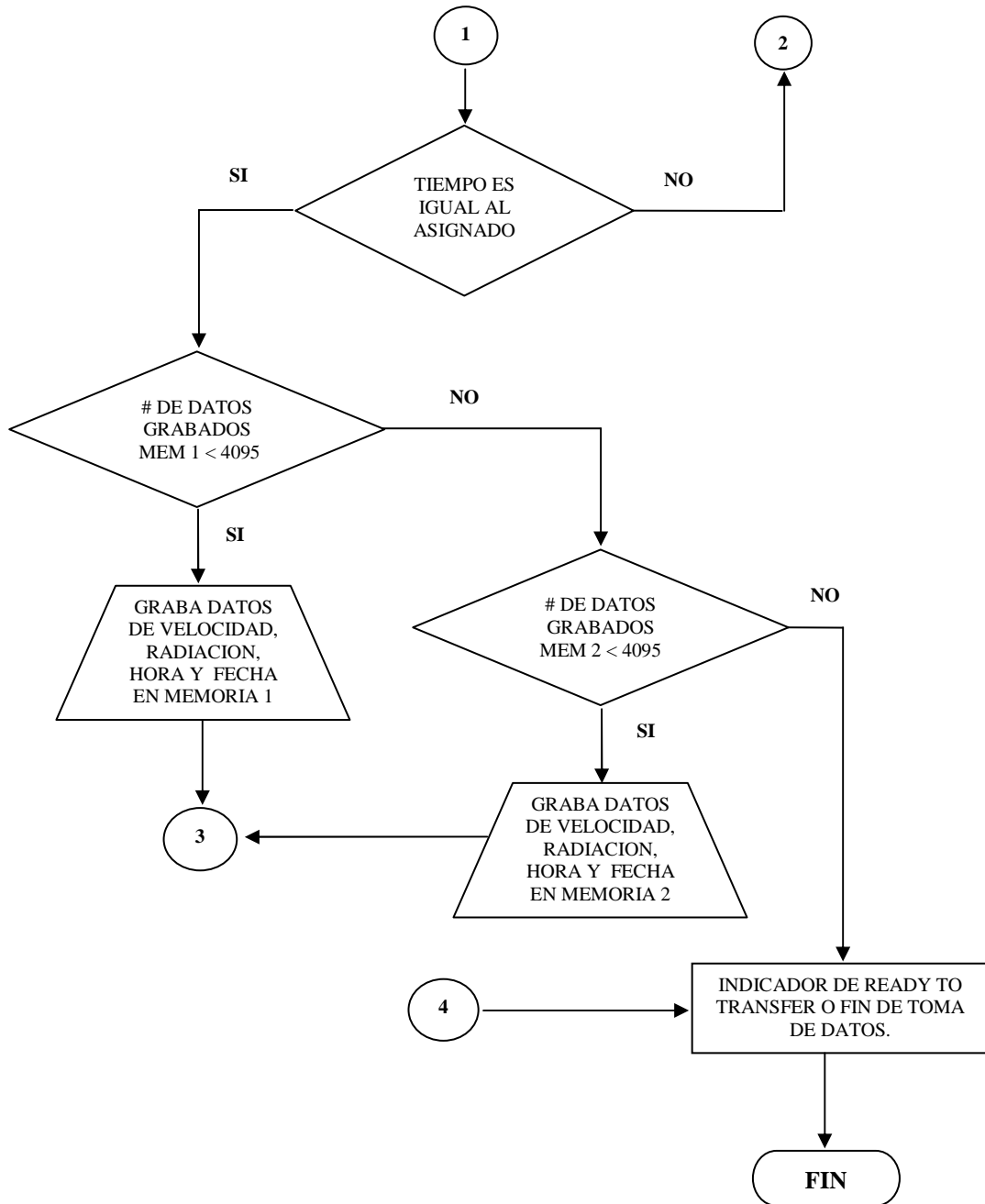


Figura. 2.7. Diagrama de flujo del Código del Programa

En el ANEXO E, se encuentra el programa del sistema de sensores detallado en su totalidad.

2.7.2. Diagrama del Circuito de Almacenamiento de Datos

En el ANEXO F, se muestra el diagrama del circuito de sensores. Este diseño está desarrollado con el Programa Protel DXP 2004, el cual a su vez, permitirá en lo posterior realizar el diseño de la placa (PCB) del circuito general.

2.8 DIAGRAMA DE BLOQUES DEL SISTEMA DE SENSORES

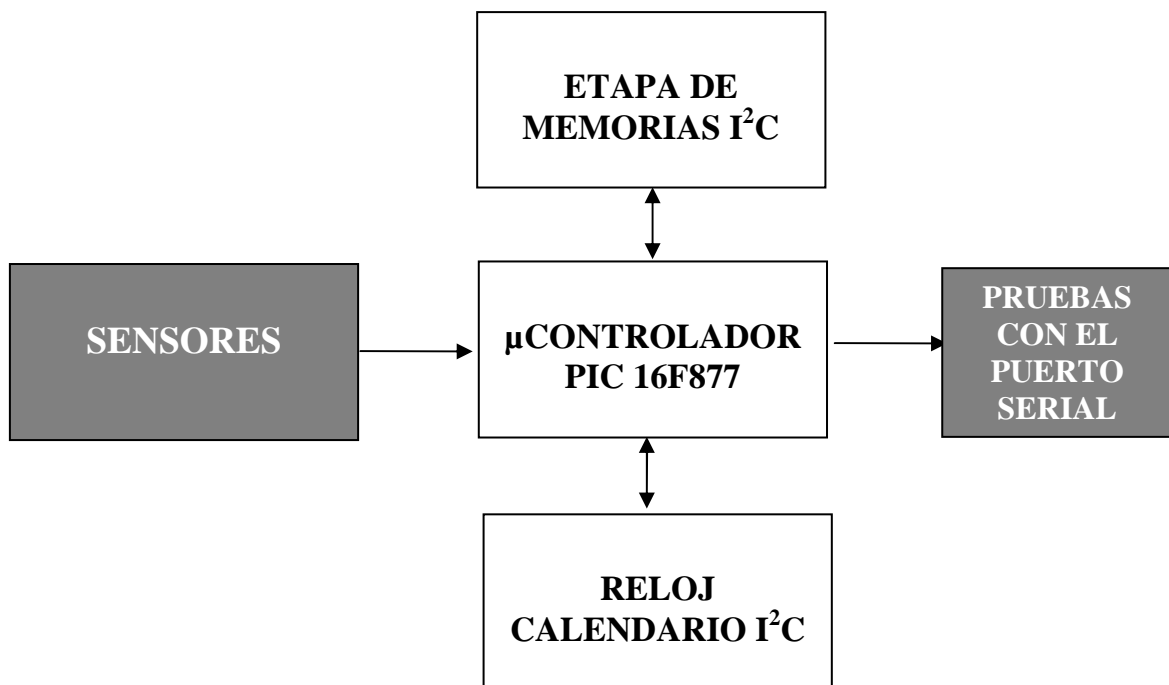


Figura. 2.8. Diagrama de Bloques del Sistema de Sensores

CAPITULO III

DISEÑO DEL SISTEMA DE MONITOREO Y ANÁLISIS EÓLICO – SOLAR

3.1 TARJETA USB DE ADQUISICIÓN DE DATOS

Para elaborar nuestra tarjeta de adquisición, utilizaremos el microcontrolador 16C745/JW el cual tiene la capacidad de conexión USB 1.1, además de las características propias de los microcontroladores de Gama Media.

En el ANEXO G, se presentan las hojas técnicas, que detallan de mejor manera el funcionamiento del PIC 16C745.



3.1.1. Introducción al Puerto USB

Hoy día resulta muy interesante observar como los avances tecnológicos nos sorprenden por la evolución tan rápida que presentan, algo que gusta es que cada vez son más fáciles de usar para cualquier persona, es decir, se están volviendo muy amigables y no se necesita ser un experto para poder comprender su funcionamiento, usarlos o instalarlos, este es el caso de Universal Serial Bus, mejor conocido como USB.

Uno de los principales propósitos, es dar a conocer la tecnología que está presente en nuestros PC's. Muy poca gente conoce las especificaciones técnicas. No existe casi tiempo para dedicar a la culturización y expansión de nuestros conocimientos, es por eso que el presente trata de orientarse a instruir a los lectores sobre esta tecnología que ya se está difundiendo masivamente.

El USB se dio a conocer debido sobre todo a la lista de tecnologías contenidas en Windows 98, es un nuevo bus que permite conectar una serie de dispositivos con una velocidad de datos que varía de baja a media-rápida.

Lo mejor de todo es el tipo de dispositivos que se pueden conectar a USB. Éstos incluyen discos duros, DVD-ROMs y CD-ROMs de alta velocidad, impresoras, escáneres, cámaras de fotos digitales, videocámaras DV, televisiones, teclados, mouse y un sin fin de aparatos. De hecho, ya hay disponibles las llamadas tarjetas de adquisición de datos USB, gracias a estos medios de comunicación se podrán conectar dispositivos como cámaras digitales y de video sin la necesidad de incómodas tarjetas externas que vienen opcionalmente con estos dispositivos y sin tener que diferenciar el puerto de conexión.

3.1.2. Universal Serial Bus (USB)

En un principio teníamos la interfaz serie y paralelo, pero era necesario unificar todos los conectores creando uno más sencillo y de mayores prestaciones. Así nació el USB (Universal Serial Bus) con una velocidad de 12 Mbps y como su evolución, USB 2.0, apodado USB de alta velocidad, con velocidades en este momento de hasta 480 Mbps, es decir, 40 veces más rápido que las conexiones mediante cables USB 1.1.

Un puerto USB puede llegar a transmitir a velocidades entre 1,5 Mbps y 12 Mbps; un puerto paralelo entre 600 Kbps a 1,5 Mbps y un puerto serial puede llegar hasta 112 Kbps. USB es una nueva arquitectura de bus o un nuevo tipo de bus desarrollado por un grupo de siete empresas (Compaq, Digital Equipment Corp., IBM PC Co., Intel, Microsoft, NEC y Northern Telecom) que forma parte de los avances plug & play y permite instalar periféricos sin tener que abrir la máquina para instalarle hardware, es decir, basta con que se conecte dicho periférico en la parte posterior del computador y listo.

3.1.2.1. ¿Qué es USB?

USB Universal Serial Bus es una interfase plug & play entre el PC y ciertos dispositivos tales como teclados, mouse, escáneres, impresoras, módems, placas de sonido, cámaras, etc.

Una característica importante es que permite a los dispositivos trabajar a velocidades mayores, en promedio a unos 12 Mbps, esto es más o menos de 3 a 5 veces más rápido que un dispositivo de puerto paralelo y de 20 a 40 veces más rápido que un dispositivo de puerto serial.

3.1.2.2. ¿Cómo funciona?

Trabaja como interfaz para transmisión de datos y distribución de energía, que ha sido introducida en el mercado de PC's y periféricos para mejorar las lentas interfaces serie (RS-232) y paralelo. Esta interfaz de 4 hilos, 12 Mbps y "plug & play", distribuye 5V para alimentación, transmite datos y está siendo adoptada rápidamente por la industria informática.

Es un bus basado en el paso de un testigo, semejante a otros buses como los de las redes locales en anillo con paso de testigo y las redes FDDI. El controlador USB distribuye testigos por el bus. El dispositivo cuya dirección coincide con la que porta el testigo responde aceptando o enviando datos al controlador. Este también gestiona la distribución de energía a los periféricos que lo requieran.

Emplea una topología de estrellas apiladas que permite el funcionamiento simultáneo de 127 dispositivos a la vez. En la raíz o vértice de las capas, está el controlador anfitrión o host que controla todo el tráfico que circula por el bus. Esta topología permite a muchos dispositivos conectarse a un único bus lógico sin que los dispositivos que se encuentran más abajo en la pirámide sufran retardo. A diferencia de otras arquitecturas, USB no es un bus de almacenamiento y envío, de forma que no se produce retardo en el transporte de un paquete de datos hacia capas inferiores. En la Fig. 3.1 se muestra la estructura de capas de USB.

El sistema de bus serie universal USB consta de tres componentes:

- Controlador
- Hubs o Concentradores
- Periféricos

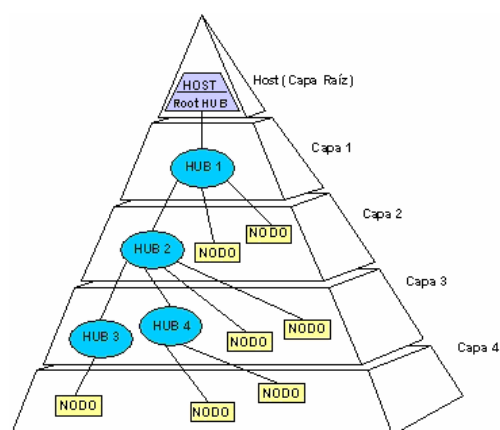


Figura. 3.1. Estructura de Capas del Bus USB

3.1.2.3. Controlador

Reside dentro del PC y es responsable de las comunicaciones entre los periféricos USB y la CPU del PC. Es también responsable de la admisión de los periféricos dentro del bus, tanto si se detecta una conexión como una desconexión. Para cada periférico añadido, el controlador determina su tipo y le asigna una dirección lógica para utilizarla siempre en las comunicaciones con el mismo. Si se producen errores durante la conexión, el controlador lo comunica a la CPU, que a su vez, lo transmite al usuario. Una vez que se ha producido la conexión correctamente, el controlador asigna al periférico los recursos del sistema que éste precise para su funcionamiento.

El controlador también es responsable del control de flujo de datos entre el periférico y la computadora.

3.1.2.4. Concentradores o Hubs

Son distribuidores inteligentes de datos y alimentación, y hacen posible la conexión a un único puerto USB de 127 dispositivos. De una forma selectiva reparten datos y alimentación hacia sus puertos descendentes y permiten la comunicación hacia su puerta de retorno o ascendente. Un hub de 7 puertos, por ejemplo, acepta datos del PC para un periférico por su puerta de retorno o ascendente y los distribuye a las 7 puertos descendentes si fuera necesario, ver Figura 3.2.

Los concentradores también permiten las comunicaciones desde el periférico hacia el PC, aceptando datos en las 4 puertas descendentes y enviándolos hacia el PC por la puerta de retorno.

Además del controlador, el PC también contiene el concentrador raíz. Este es el primer concentrador de toda la cadena que permite a los datos y a la energía pasar a uno o dos conectores USB del PC, y de allí a los 127 periféricos que, como máximo, puede soportar el sistema. Esto es posible añadiendo concentradores adicionales. Por ejemplo, si el PC tiene una única puerta USB y a ella le conectamos un hub o concentrador de 4 puertas, el PC se queda sin más puertas disponibles. Sin embargo, el hub de 4 puertas permite realizar 4 conexiones descendentes. Conectando otro hub de 4 puertas a una de las 4 puertas del primero habremos creado un total de 7 puertas a partir de una puerta del PC. De esta forma, es decir, añadiendo concentradores, el PC puede soportar hasta 127 periféricos USB.

La mayoría de los concentradores se encontrarán incorporados en los periféricos. Por ejemplo, un monitor USB puede contener un concentrador de 7 puertas incluido dentro de su chasis. El monitor utilizará una de ellas para sus datos y control y le quedarán 6 para conectar allí otros periféricos.

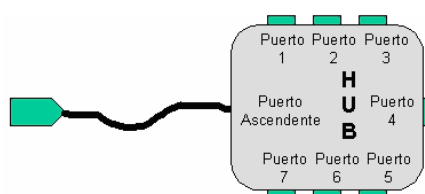


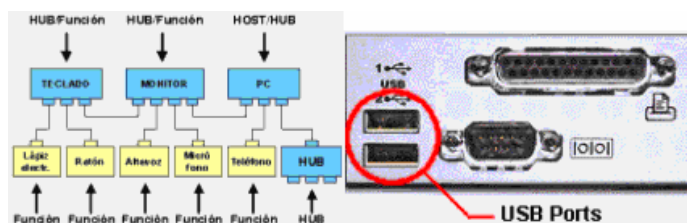
Figura. 3.2. Esquema de un Concentrador o HUB de USB

3.1.2.5. Periféricos

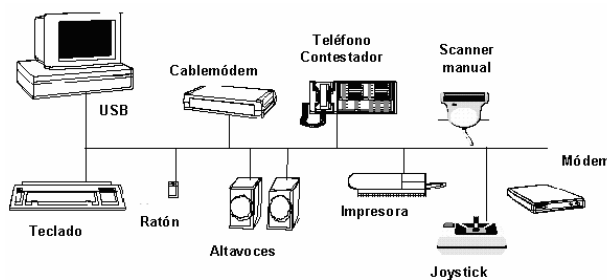
USB soporta periféricos de baja y media velocidad. Empleando dos velocidades para la transmisión de datos de 1,5 y 12 Mbps se consigue una utilización más eficiente de sus recursos. Los periféricos de baja velocidad tales como teclados, ratones, joysticks, y otros periféricos para juegos, no requieren 12 Mbps. Empleando para ellos 1,5 Mbps, se puede dedicar más recursos del sistema a periféricos tales como monitores, impresoras, módems,

escáneres, equipos de audio, que precisan de velocidades más altas para transmitir mayor volumen de datos o datos cuya dependencia temporal es más estricta.

En las Figuras 3.3 (a) y (b) se puede ver cómo los hubs proporcionan conectividad a toda una serie de dispositivos periféricos.



(a)



(b)

Figura. 3.3. (a) Posible esquema de Conexiones del USB, (b) Dispositivos USB conectados a un PC

3.1.2.6. Diagrama de capas

En el diagrama de capas de la Fig. 3.4 se observa cómo fluye la información entre las diferentes capas a nivel real y a nivel lógico.

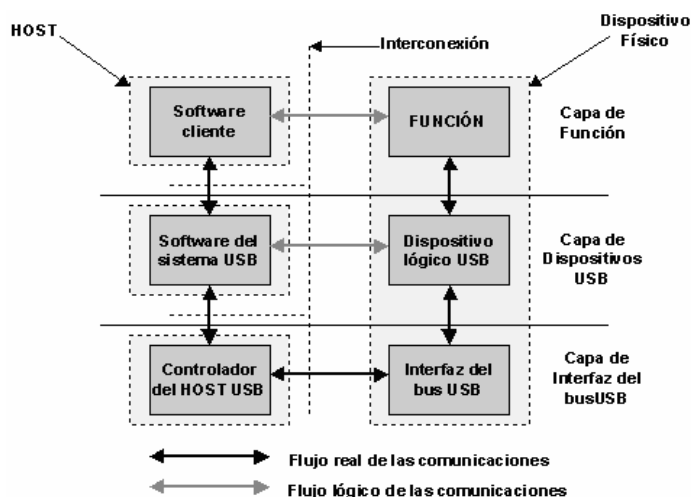


Figura. 3.4. Capas del Sistema de Comunicaciones USB

En dicha figura está materializada la conexión entre el controlador anfitrión o host y un dispositivo o periférico. Este está constituido por hardware al final de un cable USB y realiza alguna función útil para el usuario.

El software cliente se ejecuta en el host y corresponde a un dispositivo USB; se suministra con el sistema operativo o con el dispositivo USB. El software del sistema USB, es el que soporta USB en un determinado sistema operativo y se suministra con éste independientemente de los dispositivos USB o del software cliente.

El controlador anfitrión USB está constituido por el hardware y el software que permite a los dispositivos USB ser conectados al anfitrión. Como se muestra en la Figura 3.4, la conexión entre un host y un dispositivo requiere la interacción entre las capas. La capa de interfaz de bus USB proporciona la conexión física entre el host y el dispositivo.

La capa de dispositivo USB es la que permite que el software del sistema USB realice operaciones genéricas con el dispositivo.

La capa de función proporciona capacidades adicionales al host vía una adecuada capa de software cliente. Las capas de función y dispositivos USB tienen cada una de ellas una visión de la comunicación lógica dentro de su nivel, aunque la comunicación entre ellas se hace realmente por la capa de interfaz de bus USB.

3.1.2.7. Cables y conectores

USB transfiere señales y energía a los periféricos utilizando un cable de 4 hilos apantallado para transmisiones a 12 Mbps y no apantallado para transmisiones a 1,5 Mbps. En la Figura 3.5 se muestra un esquema del cable, con dos conductores para alimentación y los otros dos para señal, debiendo estos últimos ser trenzados o no, según la velocidad de transmisión.

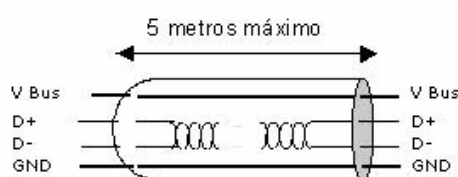


Figura. 3.5. Esquema del cable USB

El calibre de los conductores destinados a alimentación de los periféricos varía desde 20 a 26 AWG, mientras que el de los conductores de señal es de 28 AWG. La longitud máxima de los cables es de 5 metros.

Por lo que respecta a los conectores hay que decir que son del tipo ficha (o conector) y receptáculo. Estos se dividen en dos estándares, A y B y pueden ser: serie A, serie B, mini A, mini B y mini AB (Figura 3.6). Los del primer estándar presentan las cuatro patillas correspondientes a los cuatro conductores alineadas en un plano. El color recomendado es blanco sucio y los receptáculos se presentan en cuatro variantes: vertical, en ángulo recto, panel y apilado en ángulo recto así como para montaje pasamuros. Se emplean en aquellos dispositivos en los que el cable externo, está permanentemente unido a los mismos, tales como teclados, ratones, y hubs o concentradores.

Los conectores del estándar B presentan los contactos distribuidos en dos planos paralelos, dos en cada plano, y se emplean en los dispositivos que deban tener un receptáculo al cual poder conectar un cable USB. Por ejemplo, impresoras, escáneres y módems.

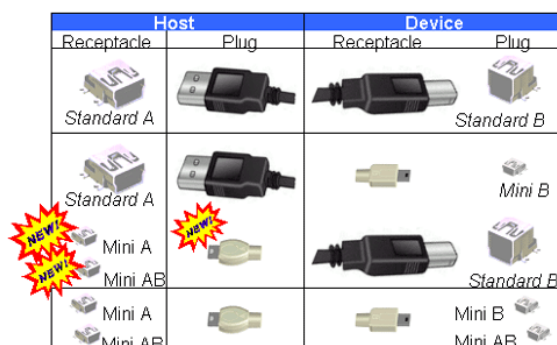


Figura. 3.6. Conectores USB

3.1.3. Aspectos Básicos del Firmware para el Microcontrolador

Microchip proporciona una capa de software para el PIC16C745/65 que maneja el interfaz a nivel más bajo. Esto provee una simple interfaz Put/Get (poner/obtener) para la comunicación. La mayor parte de procesos USB ocurren vía la Estructura de Interrupciones y estos son transparentes para la aplicación. Sin embargo todavía consume recursos de procesamiento. Del punto de vista de aplicación, el proceso de enumeración y la comunicación de datos ocurren sin la remota interacción. Sin embargo se requiere una configuración sustancial para generar descriptores apropiados.

El compilador de PicBasic Pro ahora soporta a los dispositivos USB de baja velocidad (PIC16C745 y 765).

El subdirectorio USB, instalado en el compilador contiene librerías USB de la Microchip, modificadas para PicBasic Pro, así como programas de ejemplo y algunos archivos de documentación. Los programas USB requieren que varios archivos adicionales funcionen (estos están en el subdirectorio USB), algunos de ellos requieren la modificación para su uso particular. Los archivos en este subdirectorio son:

HIDCLASS.ASM	Archivo de la Microchip HID class modificado.
JADESC.ASM	Archivo descriptor para el ejemplo.
USB_CH9.ASM	Archivo de Microchip USB CH9 modificado.
USB_DEFS.INC	Archivo Microchip USB defs modificado.
USBDESCR.ASM	Archivo Include del archive descriptor.
USBJADEM.BAS	Archivo del programa en Basic del ejemplo.

El archivo `usb_main.asm`, es útil como un punto de partida sobre una nueva aplicación y como un ejemplo de cómo una aplicación existente necesita permitir la interrupción USB y comunicarse con las funciones principales.

El archivo `usb_ch9.asm`, contiene el interfaz y las funciones principales que se necesitan para enumerar el bus. Este también contiene las funciones que atienden el USB, envían y reciben datos del host.

El archivo `hidclass.asm`, provee algunas funciones específicas HID class. Hasta ahora, sólo `Get_Report_Descriptor` y `Set_Report` son soportados. Otra clase de funciones específicas pueden ser implementadas de manera similar. Cuando se obtiene una señal de interrupción, esta nos determina que es una clase de mando específico en la que hace que el bit 6 `ReportType` sea seteado, a continuación el control pasa a la función `ClassSpecific`.

El archivo `descript.asm` contiene descriptores, configuración, end point y strings del dispositivo. Esto también incluye descriptores específicos HID (HID y reportes).

El archivo `usb_defs.inc` contiene varias macros incluyendo `ConfigUSB`, `PutEP1`, `PutEP2`, `GetEP1`, y `GetEP2`. Como se menciona antes, los casos de las macros `get` y `put` ya han sido creadas en `usb_ch9.asm`.

`HIDCLASS.ASM`, `USB_DEFS.INC` y `USB_CH9.ASM` son archivos proporcionados por la Microchip para tener acceso al hardware USB sobre el PICMICRO MCU. Estos archivos han sido modificados de modo que ellos puedan ser usados con el compilador.

El `USBJADEM`, para este caso es el “main” o programa principal, donde modificamos el programa de acuerdo a los requerimientos que se tenga. En nuestro caso particular, es aquí donde realizamos la rutina de comunicación entre el PIC 16C745 y el Programa en el PC, más adelante detallaremos el programa.

La comunicación USB es mucho más complicada que las comunicaciones sincrónicas (`Shiftin` y `Shiftout`) y asincrónicas (`Serin`, `Serout`, etcétera). Hay mucho más para saber de la operación USB que posiblemente no puede ser descrita aquí. Es por eso

que se agrega en el ANEXO H documentación más detallada, el Manual de Usuario del Firmware USB de la Microchip para Assembler.

3.1.4. Diseño del Interfaz USB

El programa para la tarjeta de adquisición de datos USB está desarrollado en MicroCode Studio, utilizando las librerías de PicBasic Pro, con lo cual se puede cargar el programa en el PIC 16C745.

Esta herramienta de trabajo, permite que se desarrolle el programa en un entorno visual, y además potencia al máximo el trabajo con microcontroladores, asignándoles, incluso, capacidades que no constan en sus especificaciones técnicas.

A continuación se presenta el diagrama de flujo del sistema de adquisición USB (Figura 3.7) que permite transferir los datos hacia una computadora.

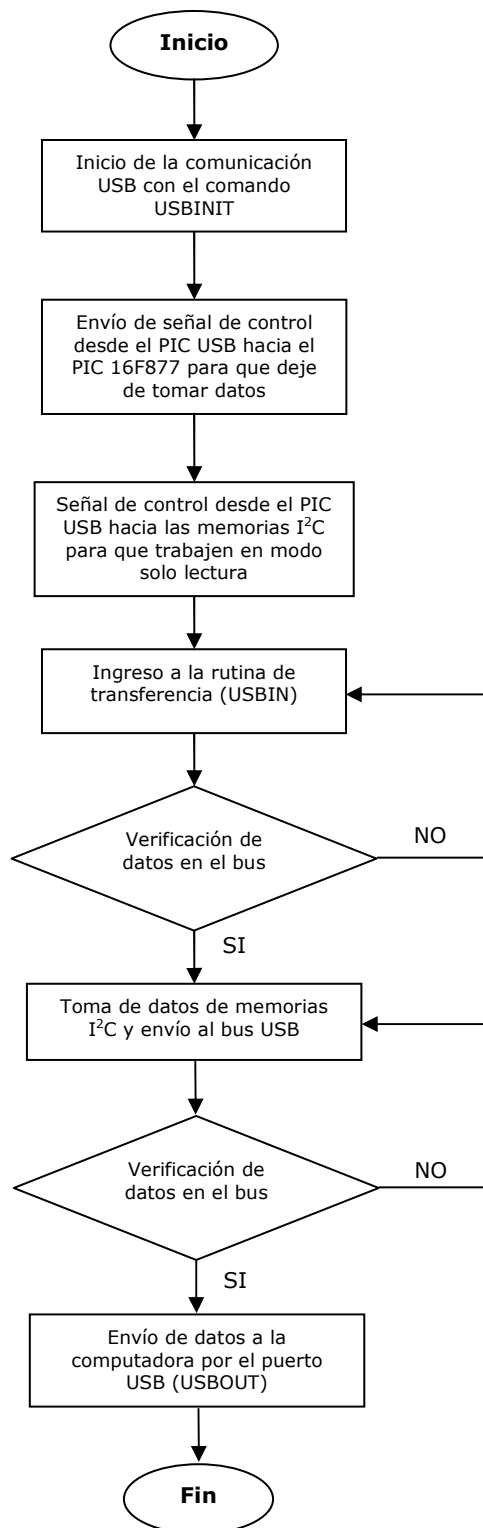


Figura. 3.7. Diagrama de flujo del Sistema de Adquisición USB

3.1.4.1. Código del Programa de la Tarjeta de Adquisición USB

El programa que se desarrolló para la tarjeta de adquisición USB consta de 5 archivos. Como se explicó anteriormente, el firmware de la microchip para los microcontroladores USB cuenta con dichos archivos.

En la práctica, se trabaja sólo en el archivo principal, en el cual se selecciona el oscilador con el que va a trabajar el PIC, se configura los puertos que se van a utilizar y se desarrolla el algoritmo general del programa.

En el ANEXO I, se detalla el código del programa que fue desarrollado para la comunicación USB.

3.1.4.2. Diagrama del Circuito de la Tarjeta de Adquisición USB

El circuito de la tarjeta de adquisición USB, es muy sencillo, ya que solo utiliza los 4 pines dedicados para la comunicación USB, así como los pines especificados para hacer comunicación I²C con las memorias y tres pines de control.

El diagrama esquemático del circuito de adquisición USB se lo realizó utilizando el programa Protel DXP 2004.

3.2 DIAGRAMA DE BLOQUES DEL SISTEMA COMPLETO

El Sistema de Monitoreo de Energía Eólica y Solar, es el resultado de la unión del sistema de sensores con el interfaz USB. Con la integración de ambos, se llevan los datos almacenados hacia la computadora, para lo cual se emplea un interfaz hombre - máquina (HMI), que permite visualizar los datos, guardarlos, modificarlos, etc.

El diagrama de bloques (Figura 3.8) que se presenta a continuación detalla como se estructuró el acoplamiento de los sistemas para almacenar los datos en la computadora.

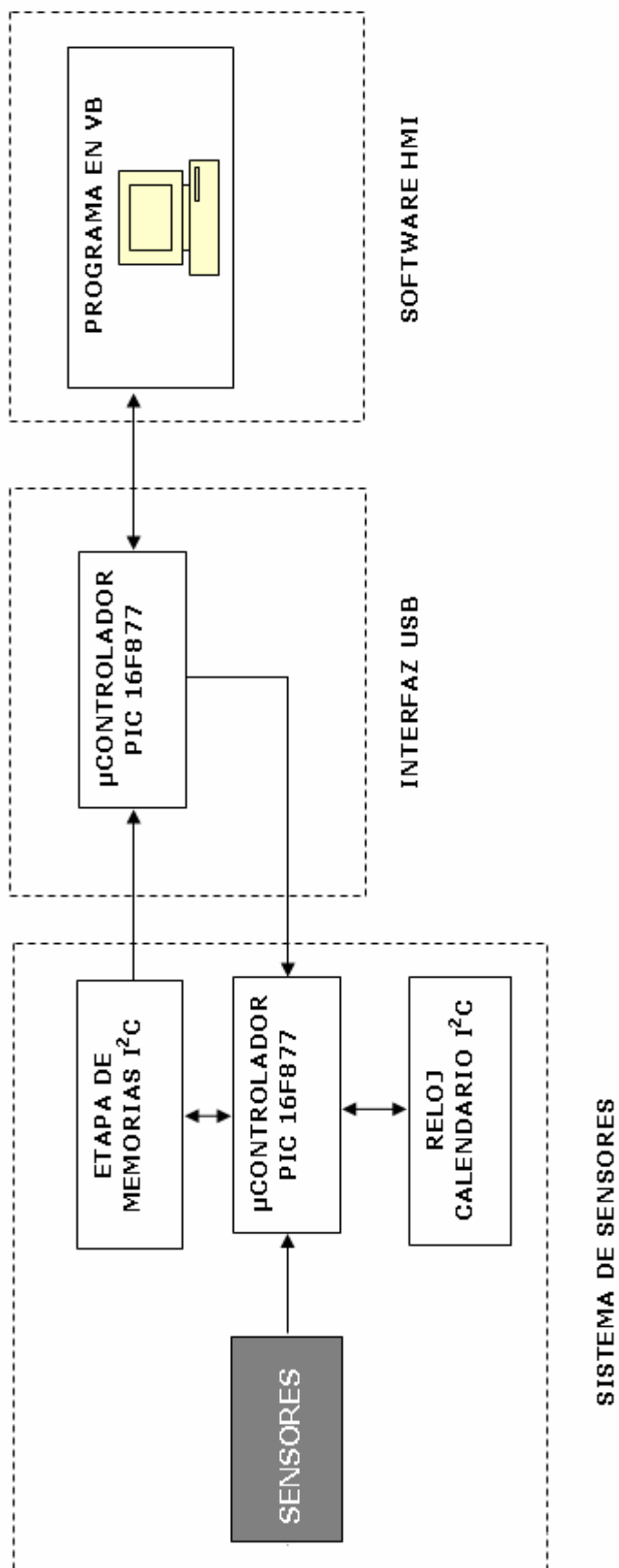


Figura. 3.8. Diagrama de Bloques del Sistema

3.2.1. Diagrama Electrónico del Circuito Total

El hardware (parte física) del Sistema de Monitoreo de Energía Eólica y Solar, es el resultado de la unión de los circuitos electrónicos de Sensores y la Tarjeta de Adquisición USB. El ANEXO J, muestra el diagrama total del sistema, el cual fue realizado en Protel DXP 2004.

3.2.2. Diseño de la Placa del Circuito (PCB)

Al igual que para el diseño esquemático del circuito, el trabajo de las placas se realizó utilizando el software Protel DXP 2004.

Este programa ayuda a los desarrolladores de circuitos a realizar placas de una calidad excepcional debido a que se basa en el esquemático para rutear los trazos y generar las pistas. El tamaño y el número de caras a ser ruteadas son configuradas en el programa, lo cual facilita el desarrollo. En el ANEXO K, se exponen las placas del sistema general.

3.3 DESARROLLO DEL HMI DE OPERACIÓN DEL SISTEMA

Para el desarrollo del Interfaz Hombre - Máquina, se utiliza el software de programación Visual Basic 6.0. Con esta herramienta de trabajo se realiza la programación de manera más sencilla y se facilita el entendimiento del programa.

Debido a que se trabaja con comunicación USB, se necesita saber que para que Visual Basic reconozca tales dispositivos, la Microchip, tiene un pequeño programa llamado HIDComm ActiveX Control.

Una vez instalado este programa, automáticamente añade un link en los componentes que utiliza Visual Basic, de manera que para poder visualizarlo es necesario dar clic derecho sobre el cuadro de herramientas de Visual Basic, buscar componentes y seleccionarlo de una lista que se despliega.

En la Figura 3.9 se muestra la manera de seleccionarlo para su adecuado uso. Una vez realizada esta acción, se presenta en el cuadro de herramientas un componente con el nombre de HID.

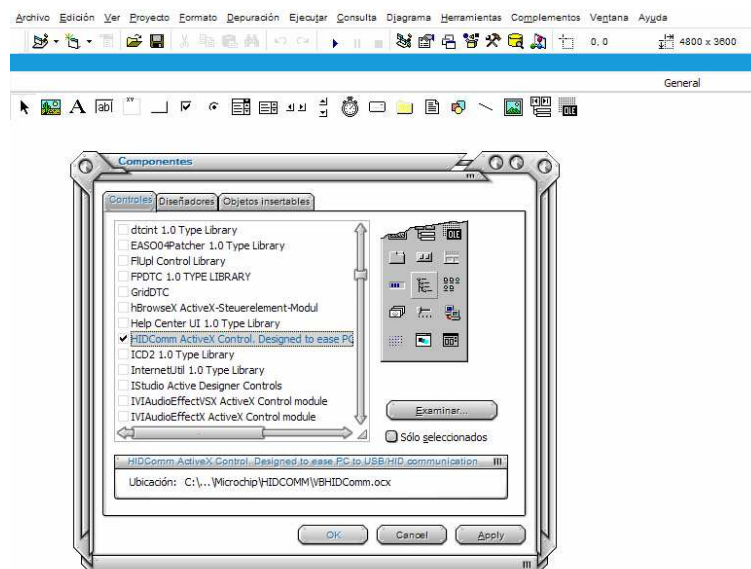


Figura. 3.9. Ventana de selección del ActiveX para comunicación USB

Una vez encontrado el HIDComm ActiveX, se procede a buscar los posibles dispositivos con los cuales interactuar. Estos dispositivos son reconocidos al estar conectados en el puerto USB, de manera que se puede encontrar mouse, teclados, impresoras, etc. y para nuestro caso particular deberá encontrar al dispositivo USB PIC 16C745. En la Figura 3.10 se muestran las ventanas en las que aparecen los dispositivos conectados al PC.

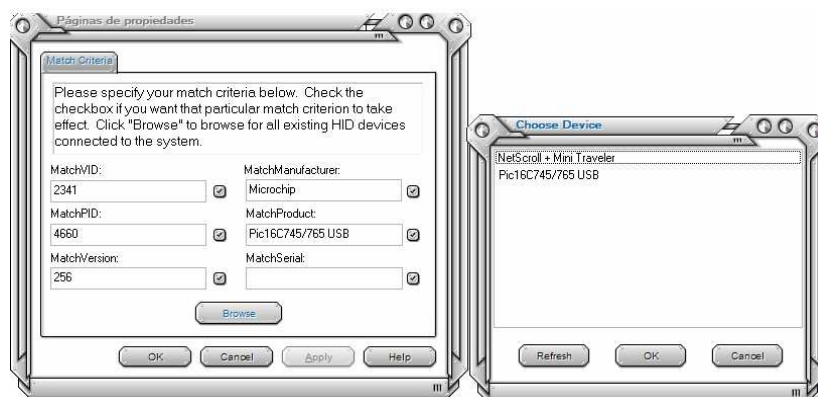


Figura. 3.10. Selección de Dispositivo USB

Después de estos pasos ya se puede desarrollar el programa de acuerdo a los objetivos planteados.

3.3.1. Desarrollo del Código de la Aplicación HMI

Para generar el código del programa, primeramente se debe realizar el diagrama de flujo de los procesos que se presentarán (Figura 3.11), de tal manera que se pueda entender de mejor forma el funcionamiento del sistema.

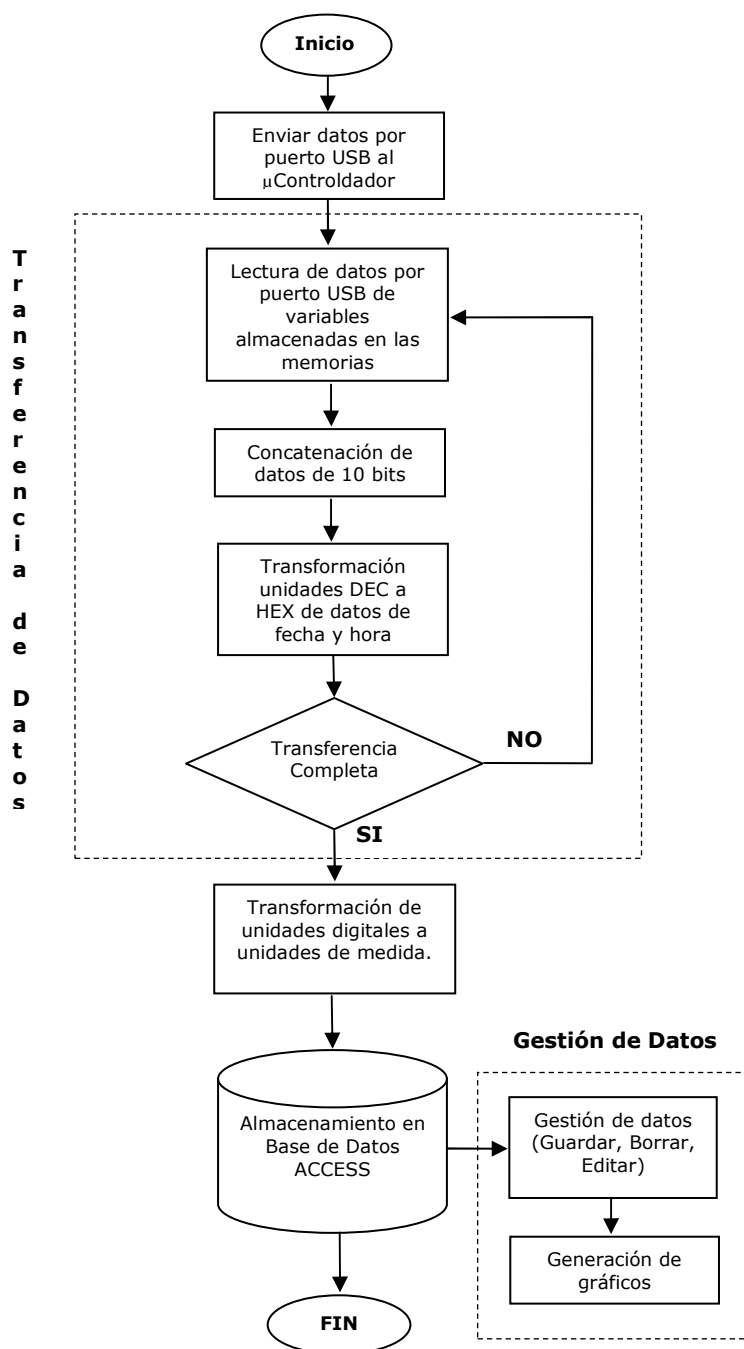


Figura. 3.11. Diagrama de flujo de los procesos de la aplicación HMI

En el ANEXO L, se presenta en detalle el código del programa utilizado para transferir y gestionar los datos del computador.

3.3.2. Programación de la Aplicación HMI

Para la programación de la interfaz hombre - máquina, se utilizó el software Visual Basic, ya que presta muchas ventajas, el lenguaje de programación es simple y fácil de aprender, su principal característica radica en el dibujado de formularios mediante el arrastre de controles. Soporta el uso de componentes COM y ActiveX. Permite crear controles personalizados fácilmente del mismo modo que el diseño de formularios.

Además se puede comunicar con otros programas de la plataforma Windows, en este caso a Microsoft Access, software diseñado para la generación de bases de datos.

A continuación se detallan las principales funciones y códigos de programación de la aplicación HMI.

Módulo

```
Public Sub Main()  
    Set CN = New ADODB.Connection  
    CN.Open "DSN=odbc_andres"  
    frm_datos.Show  
End Sub
```

De esta forma se establece la conexión a la base de datos establecida en Microsoft Access con el Middleware ODBC.

El Middleware es un software de conectividad que permite ofrecer un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas. Funciona como una capa de abstracción de software distribuida que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red).

ODBC son las siglas de Open DataBase Connectivity, que es un estándar de acceso a Bases de Datos desarrollado por Microsoft, el objetivo de ODBC es hacer posible el

acceder a cualquier dato de cualquier aplicación, sin importar qué sistema gestor de bases de datos almacene los datos.

Uno de los mecanismos que usan los programas de computadoras para comunicarse con las bases de datos, darles órdenes y obtener resultados de ellas es ADO (ActiveX Data Objects). ADO es un intermediario entre el programa y la base de datos. El programa no ve la base de datos directamente, sino que hace todo el trabajo a través de ADO. Usando ADO, el programa se comunica con la base de datos, consulta, edita, inserta, borra registros, añade tablas, etc. ADO a su vez se comunica con la base de datos a través de un "proveedor de datos".

Principales componentes de ADO:

Connection (Permite establecer una conexión con la base de datos)

Recordset (Maneja un conjunto de records de la base de datos)

Command (Permite enviar órdenes SQL para ser ejecutados por la base de datos)

El programa interfaz con el usuario, consta de varios formularios diseñados para cada tarea, a continuación se detalla el formulario principal:

Formulario 1

Con las sentencias descritas a continuación, se realiza la obtención de los datos del arreglo electrónico en intervalos de tiempo personalizados, es decir se establece la “Transferencia de Datos”.

```
Private Sub Timer1_Timer()  
If num_rec <= 20 Then  
ReDim Buffer(16)  
    Buffer(0) = CByte(2)  
    Buffer(1) = CByte(3)  
    BufferSize = 2  
    HIDComm1.WriteTo Buffer, BufferSize  
    Buffer = HIDComm1.ReadFrom(BufferSize)  
If BufferSize < 2 Then Exit Sub  
Text5(0).Text = Buffer(0)
```



```

Text5(1).Text = Buffer(1)
If aux_orden = 2 Then
If aux_orden = 1 Then
Text3.Text = deci_bina(Trim(Text5(0).Text)) + deci_bina(Trim(Text5(1).Text))
txt_vel(1).Text = Str(bina_deci(Text3.Text))
txt_realvel.Text = Format(0.07881 * Val(txt_vel(1).Text) + 0.32, "###0.00")
aux_orden = aux_orden + 1
Exit Sub
End If

```

Con la inserción del Active X HIDComm se establece la comunicación entre Visual Basic y el dispositivo USB (PIC16C745), las sentencias de lectura y escritura se reducen a estos dos commands:

```

HIDComm1.WriteTo Buffer, BufferSize
Buffer = HIDComm1.ReadFrom(BufferSize)

```

Además dentro del Formulario 1 se encuentra el procedimiento con sentencias SQL embebidas para insertar datos en la base de datos, de la siguiente manera:

```

Private Sub guardar()
If txt_rad(0).Text = "" Or txt_vel(1).Text = "" Or txt_fec(2).Text = "" Then
MsgBox "Por Favor Ingrese la Información Completa", vbInformation, "Advertencia"
Else
Dim query As String
'codigo sql
query = "INSERT INTO datos VALUES("
query = query & "" & txt_realRad.Text & ", "
query = query & "" & txt_realvel.Text & ", "
query = query & "" & txt_fec(2).Text & ""
query = query & ")"
'Ejecutando la consulta
On Error GoTo NotInserted
db.Execute query
' MsgBox "Registro Grabado Exitosamente"
Exit Sub
NotInserted:
MsgBox "Error al Grabar en BD Monitor " & Str$(Err.Number) & _

```

```
" Añadiendo Registros" & vbCrLf & _  
Err.Description  
End If  
End Sub
```

El SQL (Structured Query Language) es un lenguaje de acceso a bases de datos que explota la flexibilidad y potencia de los sistemas relacionales permitiendo gran variedad de operaciones sobre los mismos. Es un lenguaje declarativo de alto nivel, que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, y no a registros individuales, permite una alta productividad en codificación. De esta forma una sola sentencia puede equivaler a uno o más programas que utilizasen un lenguaje de bajo nivel orientado a registro.

En el caso de hacer un uso embebido del lenguaje podemos utilizar dos técnicas alternativas de programación. En una de ellas, en la que el lenguaje se denomina SQL estático, las sentencias utilizadas no cambian durante la ejecución del programa. En la otra, donde el lenguaje recibe el nombre de SQL dinámico, se produce una modificación total o parcial de las sentencias en el transcurso de la ejecución del programa. La utilización de SQL dinámico permite mayor flexibilidad y mayor complejidad en las sentencias, pero como contra punto obtenemos una eficiencia menor y el uso de técnicas de programación más complejas en el manejo de memoria y variables.

El lenguaje de Manipulación de datos, en inglés Data Manipulation Language (DML), es el que se encarga de la modificación de los datos dentro de la base de datos. Mediante este grupo de comandos, es posible consultar y modificar todos los datos de la base de datos. Es el principal componente del SQL. Existe tres operaciones básicas: INSERT, UPDATE y DELETE.

Ingresar nuevos registros

```
Insert into [tabla] values [Campos a insertar]
```

Editar Registros

```
Update [tabla] set [Campos a modificar]
```

Borrar Registros

Delete from [tabla] where [Campos a borrar]

Navegación en Datos

MoveLast

Posiciona el puntero en el último registro

MoveFirst

Posiciona el puntero en el primer registro

MoveNext

Avanza un registro

MovePrevious

Retrocede un registro

Con este tipo de sentencias se genera la Gestión de Datos, este es el Código DAO, que permite gestionar los datos de la Base de Datos. DAO (Data Access Objects) es una interfaz orientada al objeto creada por Microsoft que permite a Access y Visual Basic el uso del motor de la base de datos, además tener acceso directamente a fuentes de datos de ODBC.

3.3.3. Presentación del Interfaz Gráfico del Programa

En este capítulo se presentaran las ventanas creadas para que sirvan de interfaz gráfico y se explicará la función que desempeñan en el programa.

La primera ventana que aparece al ejecutar el programa es la carátula, que muestra los nombres de los autores del programa y el tema del proyecto. Para continuar con la presentación es requerido presionar la tecla ENTER.

En la Figura 3.12 se observa la carátula del programa.



Figura. 3.12. Carátula del Interfaz (HMI)

En la siguiente parte del programa se escoge la actividad que se desea realizar seleccionando una de las siguientes pestañas:

- Actividades
- Gráficos
- Ventana
- Ayuda

Cada una de las cuales tiene su propia subdivisión como se indica en la Figura 3.13 que se presenta a continuación.

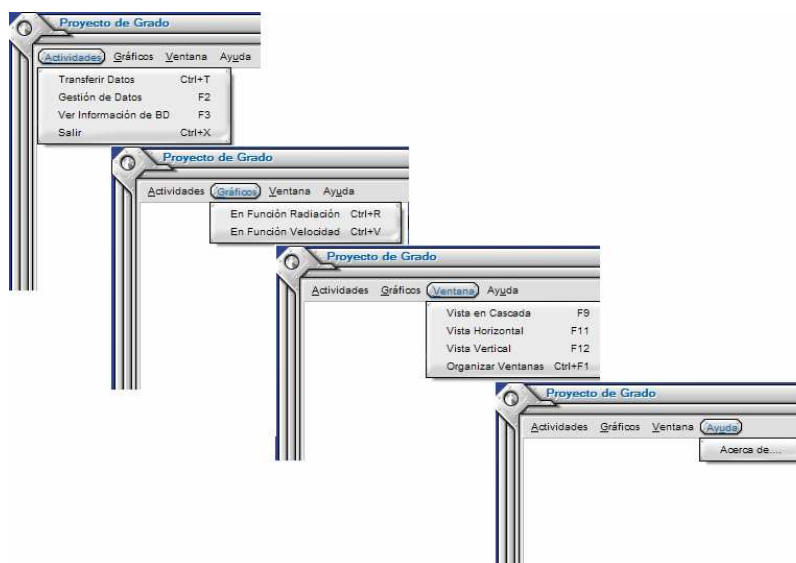


Figura. 3.13. Ventana de selección de actividades

Además, se muestran las barras de información en la parte inferior de la ventana, donde se observan dos mensajes, uno de bienvenida y el otro de información para conectar el puerto USB, así como también la hora y fecha del ordenador, como se muestra en la Figura 3.14.



Figura. 3.14. Barra de información

De estas opciones la que se prioriza es la que se encuentra en la pestaña de actividades que se llama Transferir Datos. Esta función es la que se ejecuta para que los datos sean acarreados desde las memorias hacia el computador. En la pantalla que se despliega se pueden visualizar los datos de las variables Radiación y Velocidad desde su proceso de lectura hasta su transformación en datos con unidades reales. También se observa la fecha en la que fueron tomados los datos.

A continuación se muestra la ventana Transferir Datos en la Figura 3.15.

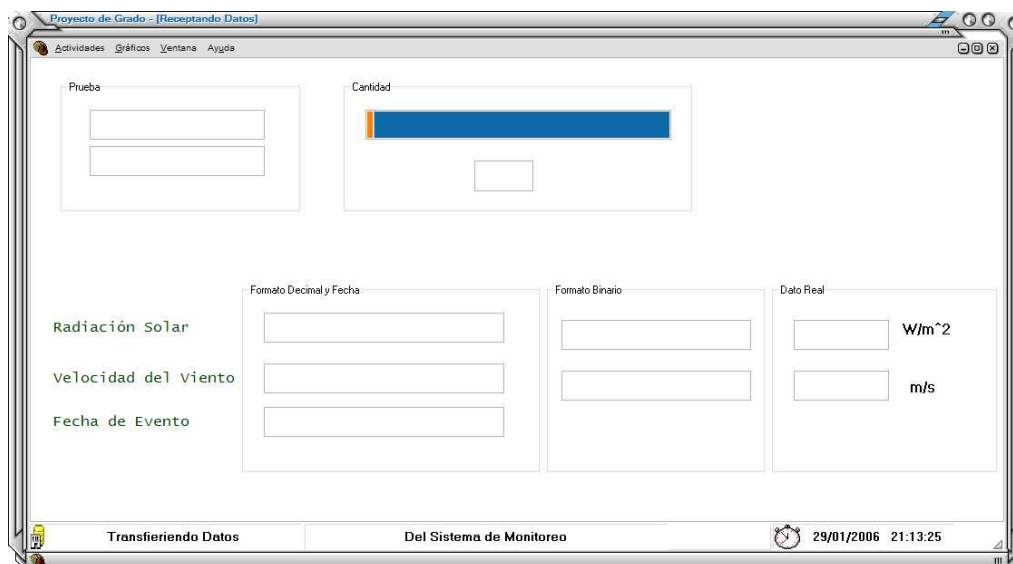


Figura. 3.15. Transferir Datos

En la opción de Gestión de Datos se encuentran funciones que permiten realizar la tarea de búsqueda, edición, navegación, consulta y gráficos. La Figura 3.16 muestra la ventana de Gestión de Datos.

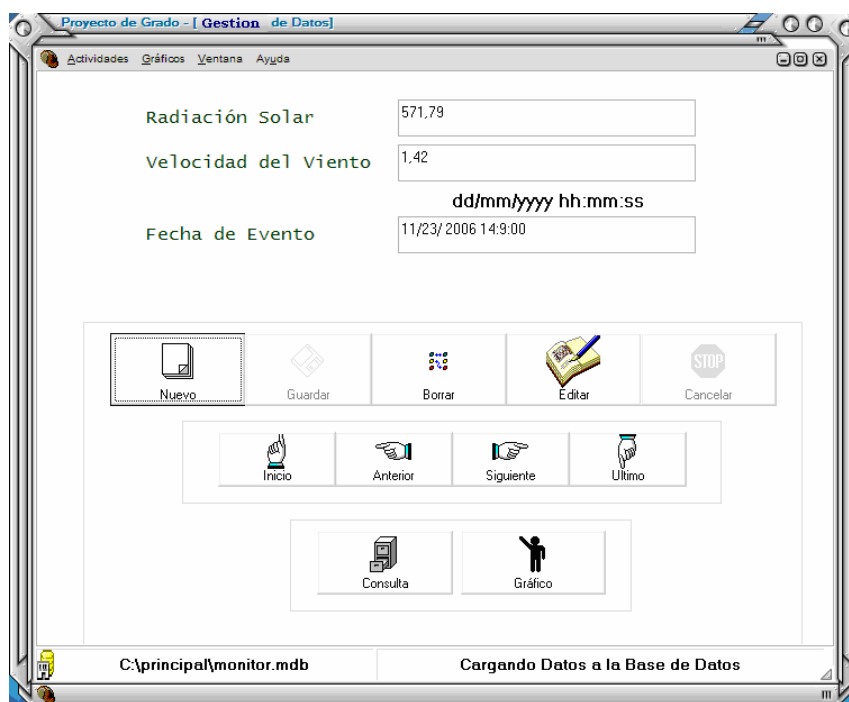


Figura. 3.16. Gestión de Datos

En las funciones restantes se puede encontrar desde la visualización de la tabla donde se encuentran almacenados los datos, hasta las gráficas de la radiación y la velocidad con respecto a la fecha y hora en la que fueron tomados los datos.

A continuación en la Figura 3.17 se muestran las ventanas mencionadas anteriormente.

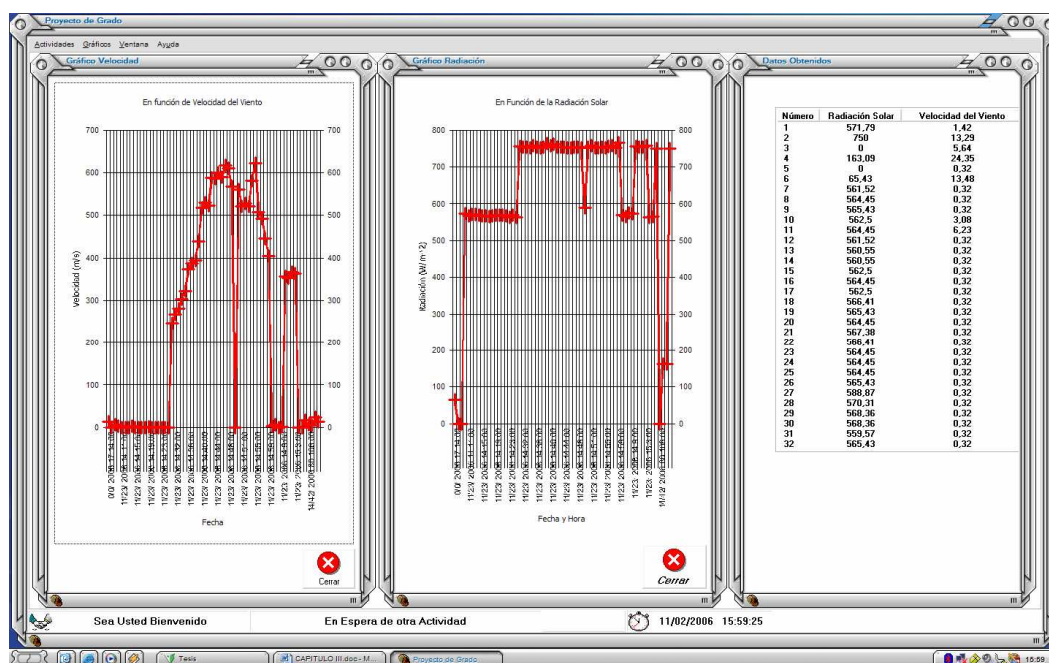


Figura. 3.17. Ventanas de Gráficos y Base de Datos

Con esto finalizaría la descripción del interfaz hombre - máquina (HMI), que fue desarrollado para poder guardar los parámetros obtenidos en la base de datos por medio de software.

3.4 ANÁLISIS DE RESULTADOS

El proceso de almacenamiento de datos es muy importante en cualquier tipo de aplicación que requiera un análisis de resultados, ya que con los datos obtenidos se puede buscar valores críticos, valores promedio, y con ello llevar un control sobre el comportamiento de los datos.

Para nuestro estudio, el análisis de resultados se lo puede realizar por medio de las gráficas que se presentaron en el HMI y obviamente también de la tabla generada por la base de datos de nuestro programa.

Antes de proceder con el análisis de los resultados presentados gráficamente se puede dar a conocer un poco de información general tanto de la radiación solar como de la velocidad del viento.

Datos sobre la Radiación Solar:

A la atmósfera llegan 1350 [W/m²]

La posición de la tierra, influye debido a la posición relativa de la tierra con el sol, ya que varía la distancia de invierno a verano, y la tierra gira sobre si misma y además varia su eje.

En la superficie de la tierra en el mejor de los casos llega 1000 [W/m²].

También influye la capa de aire, climatología, partículas en suspensión y contaminantes.

Época del año, posición geográfica.

Radiación directa y radiación difusa.

Parte de la radiación se refleja en la atmósfera.

Parte de la radiación al llegar a la tierra se refleja.

Datos sobre la Velocidad del Viento:

La velocidad del viento está siempre fluctuando, por lo que el contenido energético del viento varía continuamente.

La fluctuación del viento depende de magnitudes relacionadas tanto a las condiciones climáticas como a las de superficies locales y de los obstáculos.

En la mayoría de localizaciones del planeta el viento sopla más fuerte durante el día que durante la noche.

El viento presenta también más turbulencias y tiende a cambiar de dirección más rápidamente durante el día que durante la noche.

La información dada a conocer anteriormente sirve de guía para validar los datos que se recopilen para hacer un correcto análisis de resultados.

A continuación se presenta como ejemplo las gráficas generadas por nuestro programa tanto de la radiación solar como de la velocidad del viento para realizar un análisis de las mismas.

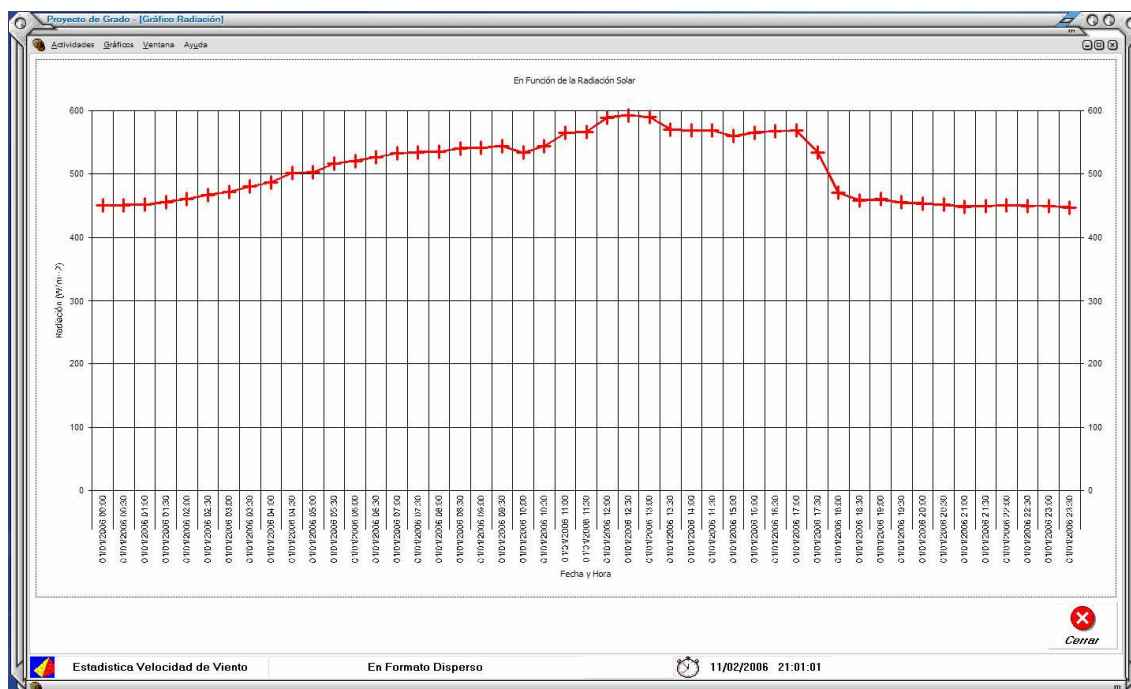


Figura. 3.18. Gráfico Radiación Vs. Tiempo

En la Figura 3.18 se puede apreciar cómo va variando en intervalos de 30 minutos la radiación solar en un día, casi despejado en Quito, específicamente, Cumbayá. De la misma forma que se tomó este dato en Quito, se lo puede realizar en todos los sectores de nuestro territorio para ir teniendo información del comportamiento de la radiación solar en nuestro país.

Teniendo en cuenta la gráfica, se puede observar el valor máximo, que en este caso particular es de $592,43 \text{ [W/m}^2\text{]}$, que se produjo a las 12:30, se puede observar el mínimo que es de $447,32 \text{ [W/m}^2\text{]}$, el cual fue a las 23:30 y realizando un promedio de todos los valores obtenidos en el día, se tiene el valor de $508,48 \text{ [W/m}^2\text{]}$.

De igual manera se presenta a continuación la gráfica de la velocidad del viento en función del tiempo para proceder al análisis (Figura 3.19).

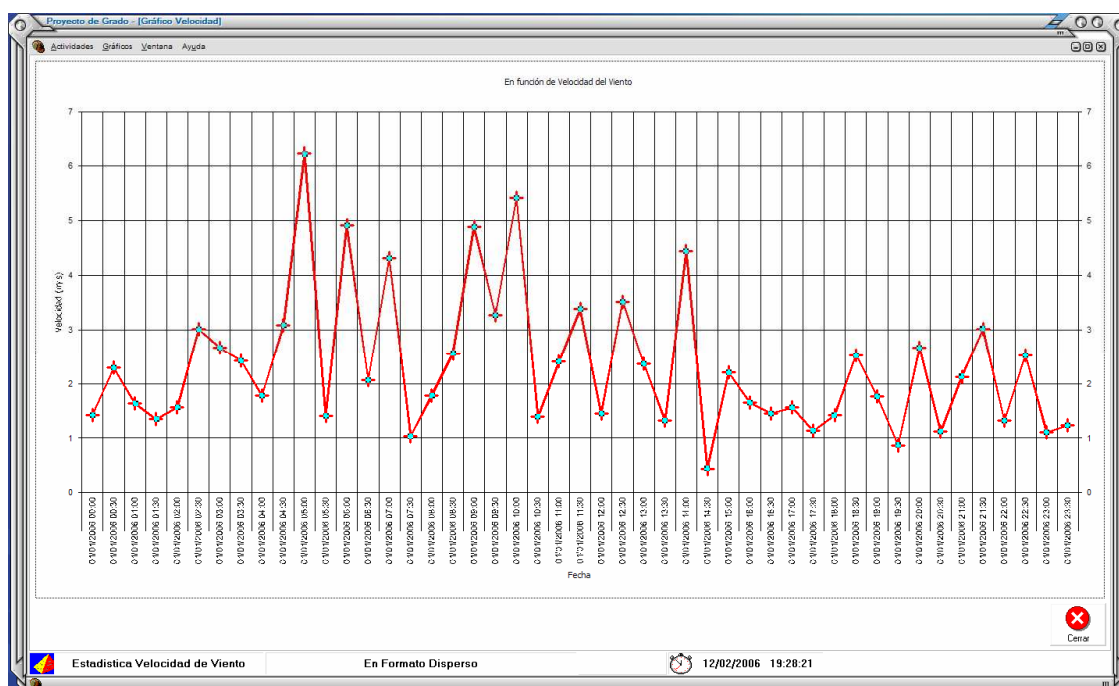


Figura. 3.19. Gráfico Velocidad del Viento Vs. Tiempo

Similar a lo que se hizo con la radiación solar, los datos de la gráfica expuesta anteriormente, fueron tomados en intervalos de 30 minutos en la ciudad de Quito. Y de lo mostrado en la gráfica anterior, se puede corroborar que la velocidad del viento tiene un comportamiento muy variable, por lo que es más crítico un seguimiento de su conducta.

Con estos datos podemos observar que el valor más alto de velocidad del viento se produjo a las 05:00, con una magnitud de 6,23 [m/s]. A las 14:30 se obtuvo el valor más bajo de velocidad de viento y fue de 0,44 [m/s], por tanto se puede obtener el promedio del día que es de 2,46 [m/s].

Con los datos presentados anteriormente, se puede realizar a diario comparaciones de sus valores para tratar de establecer similitudes o diferencias entre los distintos meses del año, entre años y así sucesivamente.

CAPITULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

Se ha cumplido el objetivo del proyecto es así que se elaboró un dispositivo portátil que mide y almacena las variables físicas de velocidad del viento y radiación del sol. Con esto finalmente se desarrolla una base de datos referente a estas variables físicas, utilizando un sistema de adquisición de datos USB.

Las fuentes de energía cada vez son más escasas en nuestro planeta, por lo que hay que buscar nuevas maneras de obtención de energía. Una de estas son las llamadas fuentes de energía renovable, como son la energía solar, la energía del viento y muchas otras más. Por este motivo se desarrolló esta tarjeta de adquisición y almacenamiento de datos con interfaz USB que nos ayudará a identificar datos reales de estas variables en nuestro medio.

Los sensores son dispositivos que transforman las variables físicas en eléctricas. Para nuestro proyecto es muy importante la selección de estos sensores, es necesario identificar la variable física a medir para adquirir el sensor adecuado. Dependiendo de las características técnicas del sensor, podremos adecuar nuestro circuito a los niveles de voltaje, corriente, frecuencia, etc. que proporcione el sensor. También se podrán obtener datos más precisos y exactos, dependiendo de su sensibilidad y límites de medición. Asimismo el costo de un sensor varía de acuerdo a la marca y características que posea.

La utilización de los microcontroladores facilita los procesos de la electrónica, brinda mayores prestaciones y minimiza la utilización de otros circuitos integrados. Por esto se escogió al microcontrolador como elemento principal en el diseño de nuestro sistema de adquisición de datos.

Para el almacenamiento de datos se escogió memorias I2C, ya que estas memorias se basan en el protocolo de comunicación serial I2C que brinda el microcontrolador. El bus I2C tiene la ventaja de utilizar solo dos líneas una para la señal de reloj y la otra para la transmisión de datos, por lo que es fácil agregar memorias en el circuito si se requiriere mayor capacidad. Además trabajan con voltajes de 2.5 a 5.5 V y la velocidad de transmisión de datos es de hasta 400 kbit/s adecuada para nuestra aplicación.

El protocolo de transmisión serial USB permite el manejo de grandes cantidades de datos a una alta velocidad. Además la interfaz USB está ganando mayores adeptos en el campo de la informática. De esta manera se puede conectar la tarjeta de adquisición de datos a cualquier PC y gestionar la información recibida de una manera sencilla y rápida.

Para el almacenamiento de datos en el PC se optó por generar bases de datos en Access programa de Microsoft ya que la cantidad de datos que se maneja es a escala media y este programa cumple con los requerimientos. Además es compatible con Visual Basic, programa en el que se desarrolló la interfaz HMI.

El uso de la interfaz HMI facilitó las tareas de almacenamiento, visualización, control y análisis de los datos transmitidos por la tarjeta de adquisición de datos USB al PC.

4.2 RECOMENDACIONES

Se debe tomar en cuenta que el PIC 16C745/JW, debe estar protegido contra los rayos del sol debido a que presenta una ventana, similar a la de las memorias EPROM, que permite borrar la información con rayos ultravioletas. Se sugiere, que se ponga un adhesivo oscuro en la ventana del PIC aún cuando éste se encuentre en el interior de una carcasa.

Si los datos que se deben tomar sobrepasan el espacio de memoria asignado, se recomienda usar memorias I2C de 512 Kb, ya que son las de mayor espacio físico de su categoría, además de que podemos conectar hasta 8 elementos de éstos al mismo bus y expandir su capacidad.

Si la información que vamos a transferir a la base de datos es demasiado extensa, se sugiere usar un software de base de datos con mayor capacidad, como el MySQL u ORACLE, los cuales presentan un mejor desempeño en manejo de bases de datos con excesiva información.

Se recomienda el uso del programa MicrocodeStudio Plus, con las librerías de Pic Basic Pro, porque presenta muchas ventajas de programación y además crea mejores recursos en microcontroladores con especificaciones limitadas. Por ejemplo, en las hojas técnicas el microcontrolador 16C745/JW no posee capacidad de comunicación I2C, pero este programa nos ayuda de una manera sencilla a asignar a dos de sus pines bidireccionales la capacidad de comunicación I2C.

Este Sistema de monitoreo de Energía Eólica y Solar con interfaz USB, se lo puede utilizar en lugares aislados o de difícil acceso, ya que es portátil, además se puede aumentar variables de temperatura, humedad relativa y pluviosidad, para determinar los lugares donde es mejor el aprovechamiento de las energías renovables.

Como el sistema fue diseñado para ser portátil y poder permanecer en lugares que presenten condiciones climáticas adversas, se debe tomar en cuenta que el encapsulado en el cual se almacene el PCB y los elementos electrónicos debe estar completamente aislado, para evitar ingreso de polvo y agua que perjudiquen su correcto funcionamiento.

En un futuro se puede ver la posibilidad de trabajar con dispositivos USB de alta velocidad, simplemente cambiando el Microcontrolador, lo que permitiría trabajar con velocidades de hasta 480 Mbps, y tomar mayor cantidad de datos en menor tiempo.

Se recomienda el uso de microcontroladores con capacidad USB para aquellas aplicaciones en las que se necesita la transmisión de datos de manera instantánea. Por ejemplo, en el caso de diseñar una tarjeta de adquisición como osciloscopio que tenga que enviar sus datos al PC, es decir aplicaciones en tiempo real.

REFERENCIAS BIBLIOGRÁFICAS

ROMERO YESA, Susana, *Microcontroladores PIC Diseño Práctico de Aplicaciones Segunda Parte PIC 16F87X*, (2da. edición), Editorial: Concesión Fernández Madrid, España, 2004, 230 páginas.

Sitios Web

www.windpower.org, Energía Eólica

www.microchip.com, Microchip

www.mecanique.co.uk, MicroCode

www.icprog.com, Icprog

www.maxim-ic.com, Dallas Semiconductor

www.Lvr.com, Jan Axelson's Lakeview Research

www.usb.org, Universal Serial Bus

www.appa.es, Asociación de Productores de Energías Renovables

www.ianai.com/ESP/energiasrenovables.htm#v, Energías Renovables

www.mysolar.com/mysolar/index.asp, My Solar

www.emagister.com, Cursos Varios

www.eolica.com.ar, Centro Regional de Energía Eólica

www.monografias.com/trabajos10/eoli/eoli.shtml#ntro, Energía Eólica

www.gurelur.org/eolica.htm, Energía Eólica Sí, pero no a cualquier precio

<http://redsolar.greenpeace.org/red.htm>, Red de Escuelas Solares

www.solarizate.org/, Aprende a Usar la Energía Solar

www.monografias.com/trabajos13/fire/fire.shtml#intro, USB

www.melabs.com/resources/samples.htm, Sample Codes

<http://es.wikipedia.org/wiki/Portada>, WIKIPEDIA, La Enciclopedia Libre.

Otros medios

Tesis “Dispositivo de Adquisición de Datos por Puerto USB” Ing. Héctor Chinchero

ANEXOS

ANEXO A
HOJA DE DATOS TÉCNICA DEL SENSOR DE VELOCIDAD DEL VIENTO

Wind Transmitter - compact



Instruction for use

4.3518.00.000 / 4.3519.00.000 / 4.3520.xx.000



General

The wind sensor measures and transmits the horizontal wind velocity. The measured values are available at the output as digital signal to control for instance wind power plant. However, with the resp. design, it can be used as input signal for dataloggers, display instruments, recording instruments or alike.

The instrument could be used in the range of 0,5...50 m/s wind velocity.

An electronically-regulated heating system has been installed for wintertime use, in order to prevent the ball bearing and the external rotation parts from freezing. Power for the heating system could be provided for instance by our **Power Supply Unit**, Order No. **9.3388.00.000**.

When using fastening adapters (angle, traverses etc.) please notice that turbulences could possibly influence the characterisitc curve

Technical Data

General

Measuring range	: 0,5 ... 50 m/s
Accuracy	: $\pm 0,5$ m/s or $\pm 3\%$ of mv.
Heating	: (Model. ...00...) 24 V AC/DC max. 20 W (Model. ...10...) non heating
Ambient temperature	: - 30 °C ... + 70 °C
Mounting	: onto mast tube (boring thread Pg 21) or onto traverse, boring \varnothing 29 mm

Order- no.

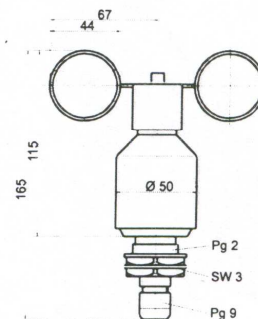
4.3518.00.000

4.3520.xx.000

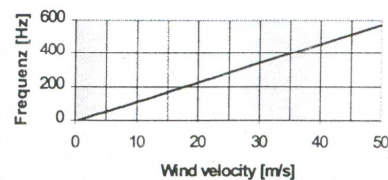
Resolution	: 10 pulse / revolution
Characteristics	: 0,5 m/s $\hat{=}$ 2 Hz ; 50 m/s = 573 Hz V [m/s] = 0,08669 • f [Hz] + 0,32

Operating voltage V_{cc}	: 12 - 24 V DC
Current input	: 20 mA
Signal output	: pulse (amplitude = \hat{V}_{cc})
Signal output load	: max. 30 mA
Scanning	: light barrier – slotted disc
Cable	: LIYCY 5 x 0,25 mm ² , 5 m long
Weight	: 0,40 kg

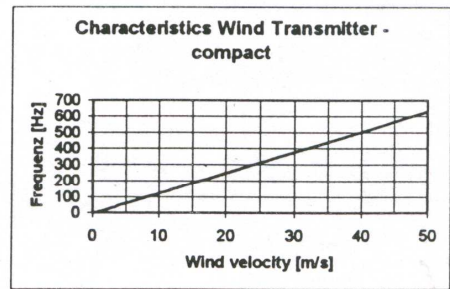
Dimension



Characteristics Wind Transmitter - compact

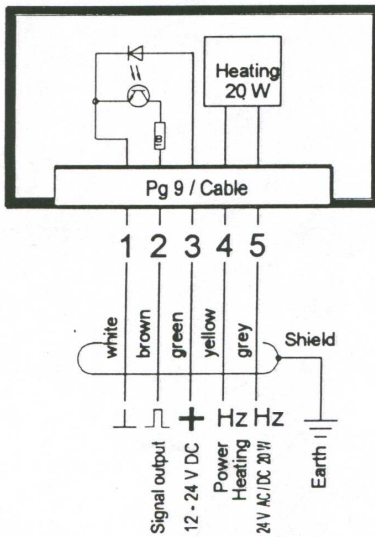


Order - no.: 4.3519.00.000 (low power)
Resolution : 11 pulse / revolution
Characteristics : 0,5 m/s $\hat{=}$ 2 Hz ; 50 m/s = 630 Hz
 $V [m/s] = 0,07881 \cdot f [Hz] + 0,32$
Operating voltage V_{cc} : 4 - 18 V DC
Current input : < 1 mA
Signal output : pulse (Amplitude = \hat{V}_{cc})
Scanning : light barrier – slot disc
Cable : LiYCY 5 x 0,25 mm², 12 m long
Weight : 0,75 kg

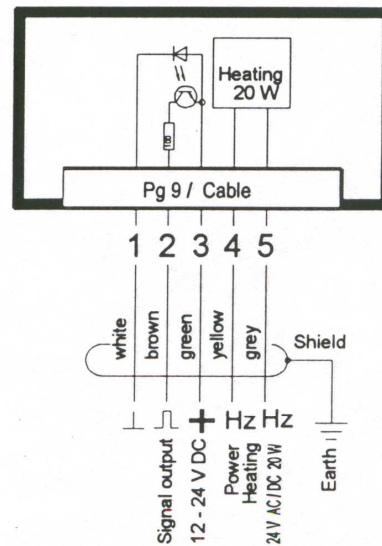


Connection diagram

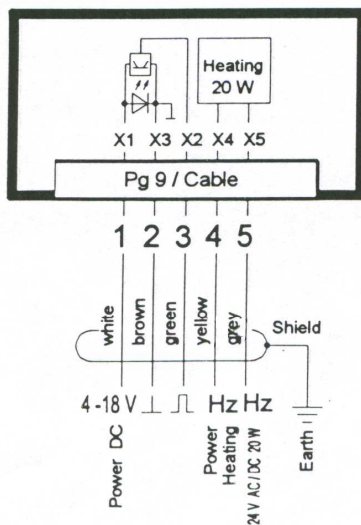
Order - no.: 4.3518.00.000



Order - no.: 4.3520.00.000



Order - no.: 4.3519.00.000



Order - no.: 4.3520.10.000

Without heating
 Pin 4 and Pin 5 not connected

Cable colour code

Cable-no.	Colour
1	white
2	brown
3	green
4	yellow
5	grey

Construction and mode of operation

The wind velocity is recorded by means of a low-inertia plastic cup star the ball-bearing axis of which is connected to a slotted disk or cup wheel. The slotted disc/cup wheel is scanned opto-electronically and supplies 10 resp. 11 pulses with every rotation (see Technical Data).

The built-in electronic forms pulses :

.....	the frequency of which is proportional to the wind velocity	order-no.:	4.3518.00.000 / 4.3520.xx.000
.....	which is used for digital data processing	order-no.:	4.3519.00.000

The external parts of the instruments are made of corrosion-resistance material (plastic) resp. the aluminium housing is additionally protected by means of an anodic coat. Labyrinth sealing protect sensitive parts inside the instrument against humidity.

Selecting a site

In general wind measurement instruments should be able to detect the wind conditions of a large area. In order to obtain comparable values when determining the surface wind, measurements should be taken at a height of 10 meters over an even area with no obstacles. An area with no obstacles means that the distance between the wind transmitter and an obstacle should be at least 10 times the height of the obstacle. If it is not possible to fulfil this condition, then the wind transmitter should be set up a height where local obstacles do not influence the measured values to any significant extent (approx. 6-10 m above the obstacle). The wind transmitter should be set up in the centre of flat roofs not on the avoid bias in the direction (privileged directions).

Mounting

The mounting of the transmitter could be done for example onto a central mast tube with a boring thread Pg 21 or on hangers with a boring of 29 mm Ø.

When using fastening adapters (angle, traverses etc.) please notice that turbulences could possibly influence the characteristic curve

After flexible connection cable is passed through the boring, wind transmitter could be fixed with hexagonal nut (WO 36). For electrical connection please refer to the connection diagram.

Maintenance

After proper mounting the instruments works maintenance free.

Heavy pollution can clog up the slit between the rotating and the stationary parts of the wind transmitter. This slit must be kept clean.



CLIMA

ADOLF THIES GmbH & Co.KG

Hauptstraße 76 37083 Göttingen Germany
Postfach 3536 + 3541 37025 Göttingen
Tel. (0551) 79001-0, Fax (0551) 79001-65
Internet <http://www.thiesclima.com> eMail info@thiesclima.com

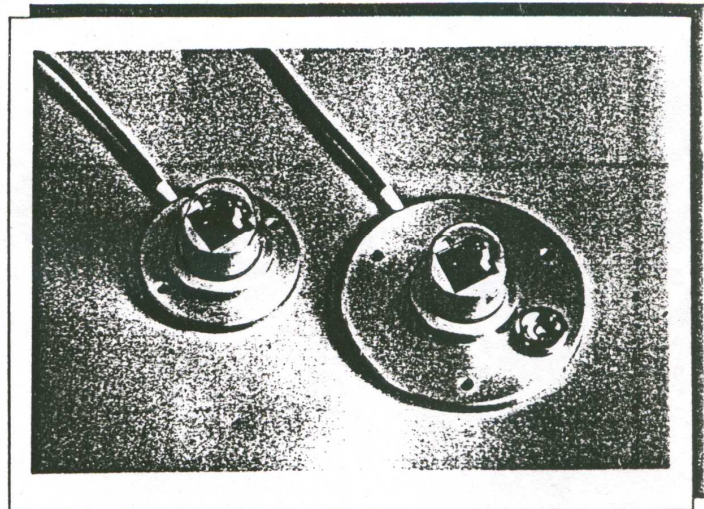


Technical Alteration reserved

ANEXO B
HOJA DE DATOS TÉCNICA DEL SENSOR DE RADIACIÓN SOLAR

Standard Silizium Radiometer

SSR 81

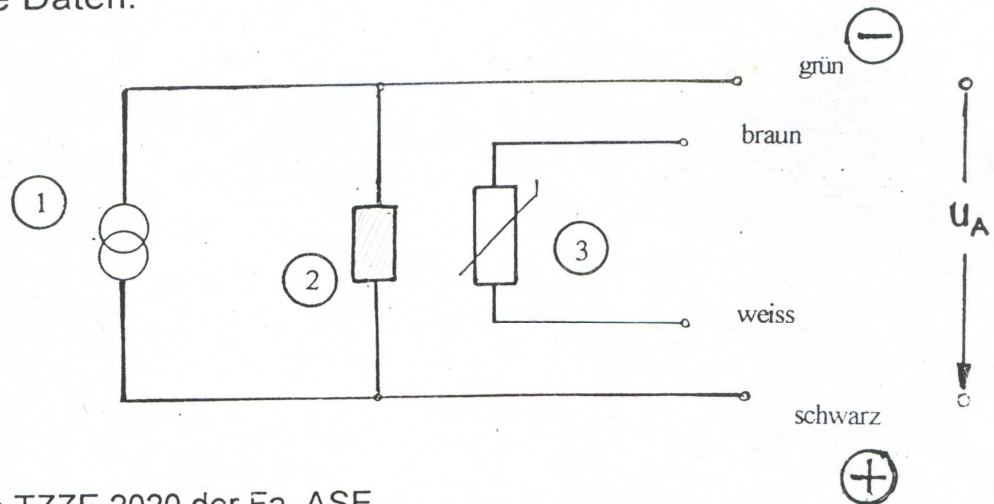


Der SSR ist ein low - cost - Sensor zur Messung von Bestrahlungsstärken in Watt pro Quadratmeter; ähnlich dem Pyranometer, liefert die unter Quarzglas wetterfest gekapselte Zelle aus monokristallinem Silizium eine Spannung proportional der Bestrahlungsstärke. Die spektrale Empfindlichkeit des Sensors entspricht dabei dem Kurvenverlauf von Silizium-Solarstrommodulen. Der Sensor, von uns werksseitig kalibriert auf einen Wert zwischen 60 mV bis 70 mV bei 1000 W/qm für ein 1,5 AM (air mass) Spektrum, liefert bei einer anderen spektralen Lichtzusammensetzung zweifellos einen vom Pyranometer merklich abweichenden Meßwert - ebenso und genauso wie die Modulen Untersuchungen haben zudem nachgewiesen, daß für Langzeit - Meßwerte (Tages- / Wochenmittel) keine signifikanten Abweichungen zwischen Pyranometer und kalibrierten Si-Zellen bestehen. Der SSR ist somit ein idealer Meßwertgeber für Bestrahlungsstärken überall dort, wo die Ausbeute von Silizium-Solarstrommodulen vergleichend gemessen werden soll - mit geringen Kosten, robust, unanfällig für Störungen. Der Sensor besteht aus einer eloxierten Alu-Scheibe (65mm Durchmesser) mit aufgesetzter Quarzkalotte als wasserdichte Abdeckung für eine sehr hochwertige, 2 x 2 cm - Solarzelle der ASE von ca. 60 mW. Die Zelle ist intern kurzgeschlossen mit einem temperaturstabilen Metallschichtwiderstand 0,06 Ohm. Außerdem ist ein Silizium-Temperatur-Sensor KTY 81 / 120 im SSR eingebaut, der es gestattet, bei Bedarf den Temperaturkoeffizienten der Zelle zu kompensieren. Die Zuleitung des KTY 81 / 120 sind getrennt ausgeführt, sodaß dieser auch unabhängig von der Zelle - zur Temperaturmessung verwendet werden kann (zwei Sensoren in einem Gehäuse). Zum Schutz vor Kondensation ist der SSR 81 mit Silicagel befüllt. Die Bauhöhe des SSR beträgt 50 mm gesamt, das 3-adrige, abgeschirmte Anschlußkabel ist bei der serienmäßigen Ausführung des SSR 5m lang, die Grundplatte ist schwarz und der Zellenträger hell eloxiert. Der SSR eignet sich vorzüglich als Geber für die Messung der Globalstrahlung und der Temperatur an den üblichen Data - Loggern / integrierenden Anzeige - Geräten, z.B. am Solar Power Meter + Integrator SIC 100.

Standard Silizium Radiometer - SSR 81

Technische Daten:

Anschlußbelegung



① Solarzelle TZZE 2020 der Fa. ASE

Solarzellenaufbau

Basismaterial: Monokristallines Silizium
 Zellenstruktur: Si n⁺ auf Si p
 Größe der Zellen: 20 mm x 20 mm
 Antireflexschicht: Titanoxid

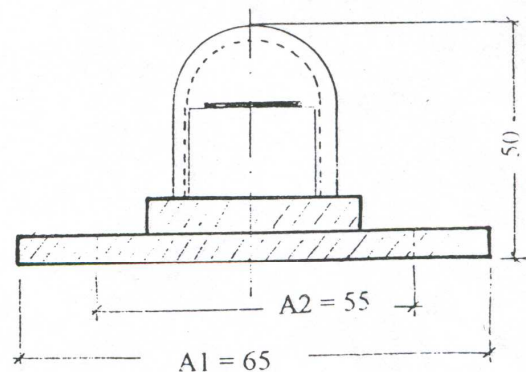
Typische el. Werte der Zelle:

$U_0 = 590 \text{ mV}$
 $I_n = 109 \text{ mA}$
 $P_{\text{max}} = 50 \text{ mW}$
 bei $A_{\text{M}} 1.5, 100 \text{ mW/cm}^2, 25^\circ\text{C}$

Temperaturkoeffizient von $I_s = +0,06\%$ pro °K

② Shunt - Widerstand

Metallschichtwiderstand 0,6 Ohm
 0,25 W, 1 %, 250 V max., TK 100



③ Temperatursensor

KTY 81/120


Fa. Siemens, n-leitender Silizium - Kristall - Widerstand

Widerstand (bei 25°C u. 1mA) 1kOhm ± 2%
 Betriebstemperatur -50/+150°C
 Thermische Zeitkonstante (Luft) 40 sec.

Betriebsgleichstrom 5mA
 Temperatur-Koeffiz. 0,79% pro °K

④ Ausgangsspannung U_A bei Bestrahlung siehe Kalibrierung auf dem Typenschild (Rückseite des Radiometers).

ANEXO C
HOJA DE DATOS TÉCNICA DE LA MEMORIA 24LC256



MICROCHIP 24AA256/24LC256/24FC256

256K I²C™ CMOS Serial EEPROM

Features

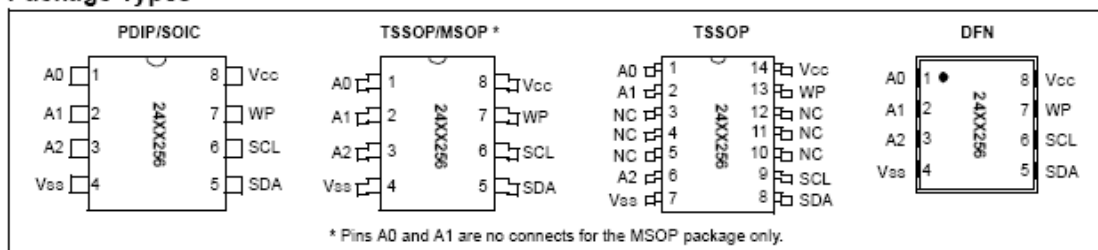
- Low power CMOS technology
 - Maximum write current 3 mA at 5.5 V
 - Maximum read current 400 μ A at 5.5 V
 - Standby current 100 nA typical at 5.5 V
- 2-wire serial interface bus, I²C compatible
- Cascadable for up to eight devices
- Self-timed ERASE/WRITE cycle
- 64-byte page-write mode available
- 5 ms max write-cycle time
- Hardware write protect for entire array
- Output slope control to eliminate ground bounce
- Schmitt trigger inputs for noise suppression
- 1,000,000 erase/write cycles
- Electrostatic discharge protection > 4000 V
- Data retention > 200 years
- 8-pin PDIP, SOIC, TSSOP, MSOP, and DFN packages
- 14-lead TSSOP package
- Temperature ranges:
 - Industrial (I): -40°C to +85°C
 - Automotive (E): -40°C to +125°C

Device Selection Table

Part Number	Vcc Range	Max. Clock Frequency	Temp. Ranges
24AA256	1.8-5.5 V	400 kHz ⁽¹⁾	I
24LC256	2.5-5.5 V	400 kHz	I, E
24FC256	2.5-5.5 V	1 MHz	I

Note 1: 100 kHz for Vcc < 2.5 V.

Package Types

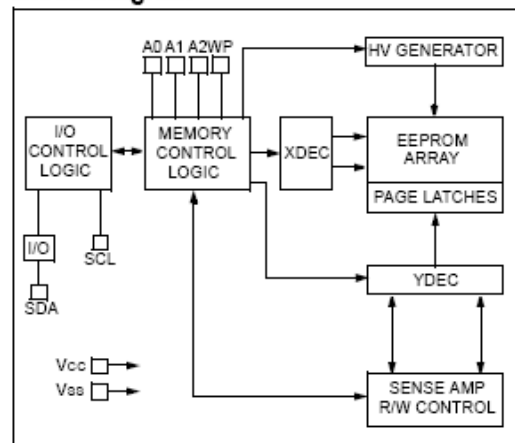


¹24XX256 is used in this document as a generic part number for the 24AA256/24LC256/24FC256 devices.

Description

The Microchip Technology Inc. 24AA256/24LC256/24FC256 (24XX256*) is a 32K x 8 (256 Kbit) Serial Electrically Erasable PROM, capable of operation across a broad voltage range (1.8 V to 5.5 V). It has been developed for advanced, low power applications such as personal communications or data acquisition. This device also has a page-write capability of up to 64 bytes of data. This device is capable of both random and sequential reads up to the 256K boundary. Functional address lines allow up to eight devices on the same bus, for up to 2 Mbit address space. This device is available in the standard 8-pin plastic DIP, SOIC, TSSOP, MSOP, DFN and 14-lead TSSOP packages.

Block Diagram



24AA256/24LC256/24FC256

1.0 ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings†

V _{CC}	6.5 V
All inputs and outputs w.r.t. V _{SS}	-0.6 V to V _{CC} +1.0 V
Storage temperature.....	-65°C to +150°C
Ambient temp. with power applied.....	-65°C to +125°C
ESD protection on all pins.....	≥ 4 kV

† NOTICE: Stresses above those listed under "Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operational listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

1.1 24AA256/24LC256/24FC256 DC Electrical Specifications

DC Specifications			Electrical Characteristics:			
			Industrial (I): V _{CC} = +1.8 V to 5.5 V		T _{AMB} = -40°C to +85°C	
			Automotive (E): V _{CC} = +2.5 V to 5.5 V		T _{AMB} = -40°C to +125°C	
Param. No.	Sym	Characteristic	Min	Max	Units	Conditions
D1	—	A0, A1, A2, SCL, SDA and WP pins:	—	—	—	—
D2	V _{IH}	High level input voltage	0.7 V _{CC}	—	V	—
D3	V _{IL}	Low level input voltage	—	0.3 V _{CC} 0.2 V _{CC}	V	V _{CC} ≥ 2.5 V V _{CC} < 2.5 V
D4	V _{HYS}	Hysteresis of Schmitt Trigger inputs (SDA, SCL pins)	0.05 V _{CC}	—	V	V _{CC} ≥ 2.5 V (Note)
D5	V _{OL}	Low level output voltage	—	0.40	V	I _{OL} = 3.0 ma @ V _{CC} = 4.5 V I _{OL} = 2.1 ma @ V _{CC} = 2.5 V
D6	I _I	Input leakage current	—	±10	μA	V _{IN} = V _{SS} or V _{CC} , WP = V _{SS} V _{IN} = V _{SS} or V _{CC} , WP = V _{CC}
D7	I _O	Output leakage current	—	±10	μA	V _{OUT} = V _{SS} or V _{CC}
D8	C _{IN} , C _{OUT}	Pin capacitance (all inputs/outputs)	—	10	pF	V _{CC} = 5.0 V (Note) T _{AMB} = 25°C, f _c = 1 MHz
D9	I _{CC} Read	Operating current	—	400	μA	V _{CC} = 5.5 V, SCL = 400 kHz
	I _{CC} Write		—	3	mA	V _{CC} = 5.5 V
D10	I _{CCS}	Standby current	—	1	μA	T _{AMB} = -40°C to +85°C SCL = SDA = V _{CC} = 5.5 V A0, A1, A2, WP = V _{SS}
			—	5	μA	T _{AMB} = -40°C to +125°C SCL = SDA = V _{CC} = 5.5 V A0, A1, A2, WP = V _{SS}

Note: This parameter is periodically sampled and not 100% tested.

24AA256/24LC256/24FC256

2.0 PIN DESCRIPTIONS

The descriptions of the pins are listed in Table 2-1.

TABLE 2-1: PIN FUNCTION TABLE

Name	8-pin PDIP	8-pin SOIC	8-pin TSSOP	14-pin TSSOP	8-pin MSOP	8-pin DFN	Function
A0	1	1	1	1	—	1	User Configurable Chip Select
A1	2	2	2	2	—	2	User Configurable Chip Select
(NC)	—	—	—	3, 4, 5	1,2	—	Not Connected
A2	3	3	3	6	3	3	User Configurable Chip Select
Vss	4	4	4	7	4	4	Ground
SDA	5	5	5	8	5	5	Serial Data
SCL	6	6	6	9	6	6	Serial Clock
(NC)	—	—	—	10, 11, 12	—	—	Not Connected
WP	7	7	7	13	7	7	Write Protect Input
Vcc	8	8	8	14	8	8	+1.8 V to 5.5 V (24AA256) +2.5 V to 5.5 V (24LC256) +2.5 V to 5.5 V (24FC256)

2.1 A0, A1, A2 Chip Address Inputs

The A0, A1 and A2 inputs are used by the 24XX256 for multiple device operations. The levels on these inputs are compared with the corresponding bits in the slave address. The chip is selected if the compare is true.

For the MSOP package only, pins A0 and A1 are not connected.

Up to eight devices (two for the MSOP package) may be connected to the same bus by using different chip select bit combinations. If these pins are left unconnected, the inputs will be pulled down internally to Vss. If they are tied to Vcc or driven high, the internal pull-down circuitry is disabled.

In most applications, the chip address inputs A0, A1 and A2 are hard-wired to logic '0' or logic '1'. For applications in which these pins are controlled by a microcontroller or other programmable device, the chip address pins must be driven to logic '0' or logic '1' before normal device operation can proceed.

2.2 Serial Data (SDA)

This is a bi-directional pin used to transfer addresses and data into and out of the device. It is an open drain terminal. Therefore, the SDA bus requires a pull-up resistor to Vcc (typical 10 k Ω for 100 kHz, 2 k Ω for 400 kHz and 1 MHz).

For normal data transfer SDA is allowed to change only during SCL low. Changes during SCL high are reserved for indicating the START and STOP conditions.

2.3 Serial Clock (SCL)

This input is used to synchronize the data transfer to and from the device.

2.4 Write Protect (WP)

This pin can be connected to either Vss, Vcc or left floating. Internal pull-down circuitry on this pin will keep the device in the unprotected state if left floating. If tied to Vss or left floating, normal memory operation is enabled (read/write the entire memory 0000-7FFF).

If tied to Vcc, WRITE operations are inhibited. Read operations are not affected.

3.0 FUNCTIONAL DESCRIPTION

The 24XX256 supports a bi-directional 2-wire bus and data transmission protocol. A device that sends data onto the bus is defined as a transmitter and a device receiving data as a receiver. The bus must be controlled by a master device which generates the serial clock (SCL), controls the bus access, and generates the START and STOP conditions while the 24XX256 works as a slave. Both master and slave can operate as a transmitter or receiver, but the master device determines which mode is activated.

24AA256/24LC256/24FC256

4.0 BUS CHARACTERISTICS

The following bus protocol has been defined:

- Data transfer may be initiated only when the bus is not busy.
- During data transfer, the data line must remain stable whenever the clock line is HIGH. Changes in the data line while the clock line is HIGH will be interpreted as a START or STOP condition.

Accordingly, the following bus conditions have been defined (Figure 4-1).

4.1 Bus not Busy (A)

Both data and clock lines remain HIGH.

4.2 Start Data Transfer (B)

A HIGH to LOW transition of the SDA line while the clock (SCL) is HIGH determines a START condition. All commands must be preceded by a START condition.

4.3 Stop Data Transfer (C)

A LOW to HIGH transition of the SDA line while the clock (SCL) is HIGH determines a STOP condition. All operations must end with a STOP condition.

4.4 Data Valid (D)

The state of the data line represents valid data when, after a START condition, the data line is stable for the duration of the HIGH period of the clock signal.

The data on the line must be changed during the LOW period of the clock signal. There is one bit of data per clock pulse.

Each data transfer is initiated with a START condition and terminated with a STOP condition. The number of the data bytes transferred between the START and STOP conditions is determined by the master device.

4.5 Acknowledge

Each receiving device, when addressed, is obliged to generate an acknowledge signal after the reception of each byte. The master device must generate an extra clock pulse which is associated with this acknowledge bit.

Note: The 24XX256 does not generate any acknowledge bits if an internal programming cycle is in progress.

A device that acknowledges must pull down the SDA line during the acknowledge clock pulse in such a way that the SDA line is stable LOW during the HIGH period of the acknowledge related clock pulse. Of course, setup and hold times must be taken into account. During reads, a master must signal an end of data to the slave by NOT generating an acknowledge bit on the last byte that has been clocked out of the slave. In this case, the slave (24XX256) will leave the data line HIGH to enable the master to generate the STOP condition.

FIGURE 4-1: DATA TRANSFER SEQUENCE ON THE SERIAL BUS

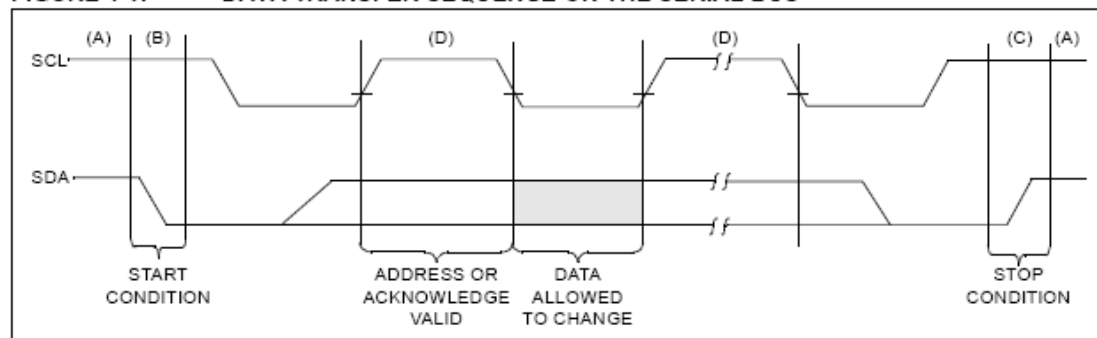
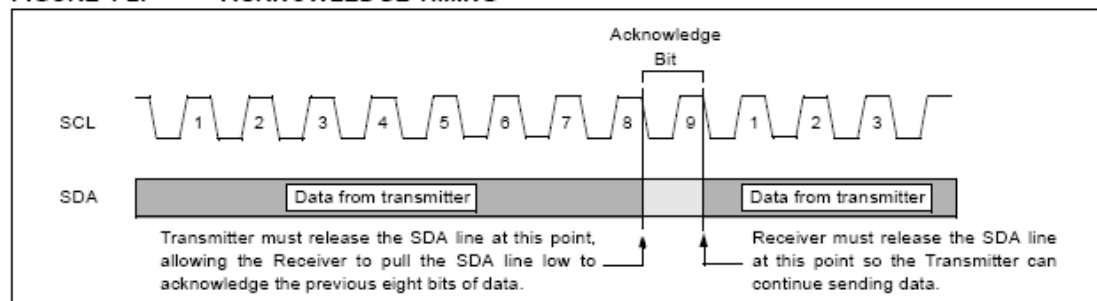


FIGURE 4-2: ACKNOWLEDGE TIMING



ANEXO D
HOJA DE DATOS TÉCNICA DEL RTC
(RELOJ CALENDARIO DE TIEMPO REAL)



DS1307 64 x 8 Serial Real-Time Clock

www.maxim-ic.com

FEATURES

- Real-time clock (RTC) counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap-year compensation valid up to 2100
- 56-byte, battery-backed, nonvolatile (NV) RAM for data storage
- Two-wire serial interface
- Programmable squarewave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500nA in battery backup mode with oscillator running
- Optional industrial temperature range: -40°C to +85°C
- Available in 8-pin DIP or SOIC
- Underwriters Laboratory (UL) recognized

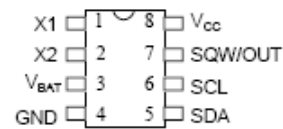
ORDERING INFORMATION

DS1307	8-Pin DIP (300-mil)
DS1307Z	8-Pin SOIC (150-mil)
DS1307N	8-Pin DIP (Industrial)
DS1307ZN	8-Pin SOIC (Industrial)

DESCRIPTION

The DS1307 Serial Real-Time Clock is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially via a 2-wire, bi-directional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power sense circuit that detects power failures and automatically switches to the battery supply.

PIN ASSIGNMENT



DS1307 8-Pin DIP (300-mil)

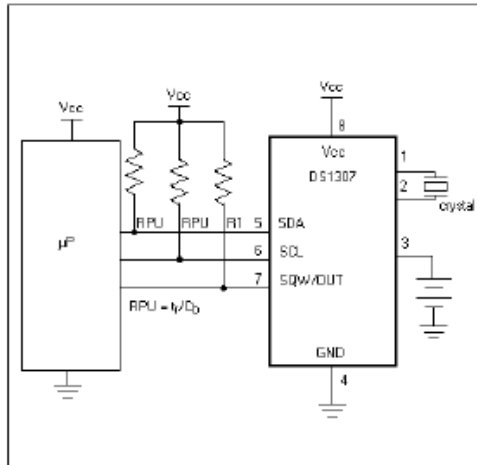


DS1307 8-Pin SOIC (150-mil)

PIN DESCRIPTION

V _{CC}	- Primary Power Supply
X1, X2	- 32.768kHz Crystal Connection
V _{BAT}	- +3V Battery Input
GND	- Ground
SDA	- Serial Data
SCL	- Serial Clock
SQW/OUT	- Square Wave/Output Driver

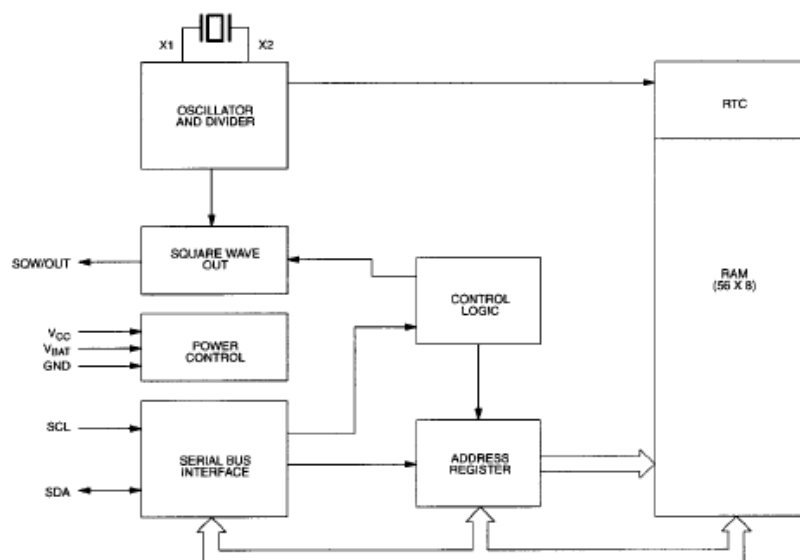
TYPICAL OPERATING CIRCUIT



OPERATION

The DS1307 operates as a slave device on the serial bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed. When V_{CC} falls below $1.25 \times V_{BAT}$ the device terminates an access in progress and resets the device address counter. Inputs to the device will not be recognized at this time to prevent erroneous data from being written to the device from an out of tolerance system. When V_{CC} falls below V_{BAT} the device switches into a low-current battery backup mode. Upon power-up, the device switches from battery to V_{CC} when V_{CC} is greater than $V_{BAT} + 0.2V$ and recognizes inputs when V_{CC} is greater than $1.25 \times V_{BAT}$. The block diagram in Figure 1 shows the main elements of the serial RTC.

DS1307 BLOCK DIAGRAM Figure 1



SIGNAL DESCRIPTIONS

V_{CC}, GND – DC power is provided to the device on these pins. V_{CC} is the +5V input. When 5V is applied within normal limits, the device is fully accessible and data can be written and read. When a 3V battery is connected to the device and V_{CC} is below 1.25 x V_{BAT}, reads and writes are inhibited. However, the timekeeping function continues unaffected by the lower input voltage. As V_{CC} falls below V_{BAT} the RAM and timekeeper are switched over to the external power supply (nominal 3.0V DC) at V_{BAT}.

V_{BAT} – Battery input for any standard 3V lithium cell or other energy source. Battery voltage must be held between 2.0V and 3.5V for proper operation. The nominal write protect trip point voltage at which access to the RTC and user RAM is denied is set by the internal circuitry as 1.25 x V_{BAT} nominal. A lithium battery with 48mAh or greater will back up the DS1307 for more than 10 years in the absence of power at 25°C. UL recognized to ensure against reverse charging current when used in conjunction with a lithium battery.

See “Conditions of Acceptability” at <http://www.maxim-ic.com/TechSupport/QA/ntrl.htm>.

SCL (Serial Clock Input) – SCL is used to synchronize data movement on the serial interface.

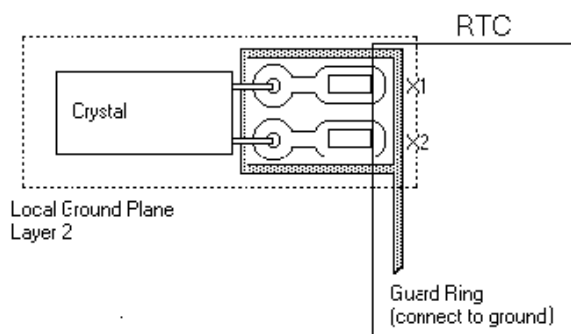
SDA (Serial Data Input/Output) – SDA is the input/output pin for the 2-wire serial interface. The SDA pin is open drain which requires an external pullup resistor.

SQW/OUT (Square Wave/Output Driver) – When enabled, the SQWE bit set to 1, the SQW/OUT pin outputs one of four square wave frequencies (1Hz, 4kHz, 8kHz, 32kHz). The SQW/OUT pin is open drain and requires an external pull-up resistor. SQW/OUT will operate with either V_{cc} or V_{bat} applied.

X1, X2 – Connections for a standard 32.768kHz quartz crystal. The internal oscillator circuitry is designed for operation with a crystal having a specified load capacitance (CL) of 12.5pF.

For more information on crystal selection and crystal layout considerations, please consult Application Note 58, “Crystal Considerations with Dallas Real-Time Clocks.” The DS1307 can also be driven by an external 32.768kHz oscillator. In this configuration, the X1 pin is connected to the external oscillator signal and the X2 pin is floated.

RECOMMENDED LAYOUT FOR CRYSTAL



CLOCK ACCURACY

The accuracy of the clock is dependent upon the accuracy of the crystal and the accuracy of the match between the capacitive load of the oscillator circuit and the capacitive load for which the crystal was trimmed. Additional error will be added by crystal frequency drift caused by temperature shifts. External circuit noise coupled into the oscillator circuit may result in the clock running fast. See Application Note 58, "Crystal Considerations with Dallas Real-Time Clocks" for detailed information.

Please review Application Note 95, "Interfacing the DS1307 with a 8051-Compatible Microcontroller" for additional information.

RTC AND RAM ADDRESS MAP

The address map for the RTC and RAM registers of the DS1307 is shown in Figure 2. The RTC registers are located in address locations 00h to 07h. The RAM registers are located in address locations 08h to 3Fh. During a multi-byte access, when the address pointer reaches 3Fh, the end of RAM space, it wraps around to location 00h, the beginning of the clock space.

DS1307 ADDRESS MAP Figure 2

00H	SECONDS
	MINUTES
	HOURS
	DAY
	DATE
	MONTH
	YEAR
07H	CONTROL
08H	RAM
3FH	56 x 8

CLOCK AND CALENDAR

The time and calendar information is obtained by reading the appropriate register bytes. The RTC registers are illustrated in Figure 3. The time and calendar are set or initialized by writing the appropriate register bytes. The contents of the time and calendar registers are in the BCD format. Bit 7 of register 0 is the clock halt (CH) bit. When this bit is set to a 1, the oscillator is disabled. When cleared to a 0, the oscillator is enabled.

Please note that the initial power-on state of all registers is not defined. Therefore, it is important to enable the oscillator (CH bit = 0) during initial configuration.

The DS1307 can be run in either 12-hour or 24-hour mode. Bit 6 of the hours register is defined as the 12- or 24-hour mode select bit. When high, the 12-hour mode is selected. In the 12-hour mode, bit 5 is the AM/PM bit with logic high being PM. In the 24-hour mode, bit 5 is the second 10 hour bit (20-23 hours).

On a 2-wire START, the current time is transferred to a second set of registers. The time information is read from these secondary registers, while the clock may continue to run. This eliminates the need to re-read the registers in case of an update of the main registers during a read.

ANEXO E
CÓDIGO DEL PROGRAMA DEL SISTEMA DE SENSORES

```

*****
* Name   : SISTEMA DE SENSORES.BAS           *
* Author : ANDRES VALLEJO y JUAN SERRANO     *
* Notice : Copyright (c) 2005 []            *
*         : All Rights Reserved              *
* Date   : 23/11/2005                        *
* Version : 1.1                               *
* Notes  : TESIS MONITOREO                   *
*         :                                  *
*****

```

```

===== CONFIGURACION OSCILADOR =====

```

```

DEFINE OSC 20    'Oscilador de 20 MHz (HS -> High Speed)

```

```

===== FIN CONFIGURACION OSCILADOR =====

```

```

===== CONFIGURACION CONVERTOR =====

```

```

DEFINE ADC_BITS 10    'Seteo número de bits de respuesta
DEFINE ADC_CLOCK 3    'Set clock source (rc = 3)
DEFINE ADC_SAMPLEUS 50 'Seteo tiempo de muestreo en microsegundos

```

```

===== FIN CONFIGURACION CONVERTOR =====

```

```

===== CONFIGURACION MEMORIA I2C =====

```

```

DEFINE I2C_HOLD 1

```

```

ID    CON %10100000 'ID de la memoria 1 I2C
ID2   CON %10100010 'ID de la memoria 2 I2C
dir_mem    var word
dir_mem_2  var word
dir_mem_aux var word
dir_mem_aux2 var word
adr_mem    var word
adr_mem_2  var word
adr_read   var word
adr_read2  var word
aux1       var WORD
aux2       var WORD
aux3       var BYTE
aux4       var BYTE
aux5       var BYTE
aux6       var byte

```

```

===== FIN CONFIGURACION MEMORIA I2C =====

```

```

===== CONFIGURACION PUERTOS =====

```

```

TRISA.0=1    'Configuro puerto A0 como entrada
ADCON1=%10001110 'Configura puerto A0 como análogo y A1, A2, A3, A5 y puerto
                'E como digitales Y CONFIGURO JUSTIFICACION A LA DERECHA
                'bit7=1 JUSTIFICACION DERECHA

```

```

TRISC.6=0    'SETEO EL PUERTO C6 COMO SALIDA (TX DE COMUNICACION SERIAL)
PORTC=%11110000 'El PORTC inicia con los displays apagados

```

```

===== FIN CONFIGURACION PUERTOS =====

===== ALIAS PARA LINEAS DE PUERTO =====

VELOC VAR PORTA.1  'Asigno alias al puerto A0
LEDV  VAR PORTB.4  'Led Verde
low LEDV
TX VAR PORTC.6    'Asigno alias al puerto C6

===== FIN DE ALIAS PARA LINEAS DE PUERTO =====

===== CONFIGURACION I2C =====

define I2C_SCLOUT 1  'Instrucción para evitar la necesidad de colocar resis-
                    'tencia pull-up en SCL
CLOCKPIN VAR PORTA.2  'Pin para señal de clock I2C
DATAPIN  VAR PORTA.3  'Pin para señal de datos I2C

===== FIN DE CONFIGURACION I2C =====

===== VARIABLES AUXILIARES =====

i var byte          'Variable para generar ciclo FOR
flag_grabacion var bit
flag_grabacion=0    'Flag para controlar la grabación del RTC
j var byte
'm var byte[3]
speed var word
SUN VAR WORD

----- VARIABLES PARA INICIALIZACION RTC -----

segundos  var byte  'Variable para almacenar segundos
minutos   var byte  'Variable para almacenar minutos
hora      var byte  'Variable para almacenar hora
num_dia   var byte  'Variable para almacenar #día (D=1,L=2,M=3,M=4,J=5
                    'V=6,S=7)
dia_fecha var byte  'Variable para almacenar #día para la fecha del mes
mes       var byte  'Variable para almacenar mes 1-12
anio     var byte  'Variable para almacenar año 00-99

----- FIN VARIABLES PARA INICIALIZACION RTC -----

----- VARIABLE DE TRANSFERENCIA -----

===== CONFIRMACION ARRANQUE =====

gosub indicador

===== FIN CONFIRMACION ARRANQUE =====

===== INICIO MEMORIA EEPROM =====

WRITE 0,0 'Dirección 0 de la memoria EEPROM se graba con 0

```

```
'===== FIN INICIO MEMORIA EEPROM ====='
```

```
***** PROGRAMA PRINCIPAL *****
```

```
inicio:
```

```
SEROUT2 TX,84,["INICIO DE PROGRAMA ",13,10]
pause 20
```

```
dir_mem =0
dir_mem_aux=0
dir_mem_aux2=0
dir_mem_2=0
adr_mem =0
adr_mem_2=0
adr_read=0
READ 0,flag_grabacion
if flag_grabacion==0 then
  gosub grabar_rtc
  WRITE 0,1 'Graba el 1 en la dirección 0 de memoria EEPROM
endif
```

```
TRISB.5=1      'Puerto A.2 declarado como entrada
TRISB.4=1
TRANSFER var PORTB.5
TRANSFER2 VAR PORTB.3
```

```
'----- FIN VARIABLE DE TRANSFERENCIA-----'
```

```
'===== FIN VARIABLES AUXILIARES ====='
```

```
TOMA_DATO:
```

```
I2CREAD DATAPIN,CLOCKPIN,% 11010000,0,[segundos]
I2CREAD DATAPIN,CLOCKPIN,% 11010000,1,[minutos]
I2CREAD DATAPIN,CLOCKPIN,% 11010000,2,[hora]
I2CREAD DATAPIN,CLOCKPIN,% 11010000,3,[num_dia]
I2CREAD DATAPIN,CLOCKPIN,% 11010000,4,[dia_fecha]
I2CREAD DATAPIN,CLOCKPIN,% 11010000,5,[mes]
I2CREAD DATAPIN,CLOCKPIN,% 11010000,6,[anio]
```

```
pause 20
```

```
if TRANSFER==1 then goto TRANSFERIR
IF TRANSFER2==1 THEN GOTO TRANSFERIR2
IF (segundos==25) OR (segundos==55) then GRABAR_DATO
```

```
GOTO TOMA_DATO:
```

```
'----- GRABAR DATO DE SENSORES -----'
```

```
GRABAR_DATO:
```

```
if adr_mem<=4095 then
```

```
SEROUT2 TX,84,["CICLO GRABADO MEMORIA 1 ",13,10]
pause 20
```

```
SEROUT2 TX,84,[" HORA: ",HEX2 hora," : ",HEX2 minutos," FECHA: ",HEX2 dia_fecha," Direccion
",13,10]
pause 20
```

```
GOSUB lectura_speed
  i2cwrite DATAPIN,CLOCKPIN,ID,dir_mem,[speed]
  dir_mem=dir_mem+2
pause 20
```

```
GOSUB solar_sensing
  I2cwrite DATAPIN,CLOCKPIN,ID,dir_mem,[SUN]
  dir_mem=dir_mem+2
pause 20
```

```
i2cwrite DATAPIN,CLOCKPIN,ID,dir_mem,[minutos]
dir_mem=dir_mem+1
pause 20
```

```
i2cwrite DATAPIN,CLOCKPIN,ID,dir_mem,[hora]
dir_mem=dir_mem+1
pause 20
```

```
i2cwrite DATAPIN,CLOCKPIN,ID,dir_mem,[dia_fecha]
dir_mem=dir_mem+1
pause 20
```

```
i2cwrite DATAPIN,CLOCKPIN,ID,dir_mem,[mes]
dir_mem=dir_mem+1
pause 20
```

```
SEROUT2 TX,84,[" DATOS GRABADOS ",13,10]
pause 500
```

```
adr_mem=adr_mem+1
```

```
goto TOMA_DATO
```

```
'===== FIN GRABADO MEMORIA 1 ====='
```

```
else
```

```
'===== GRABADO MEMORIA 2 ====='
```

```
SEROUT2 TX,84,["CICLO GRABADO MEMORIA 2 ",13,10]
pause 20
```

```
GOSUB lectura_speed
  i2cwrite DATAPIN,CLOCKPIN,ID2,dir_mem,[speed]
  dir_mem_2=dir_mem_2+2
pause 20
```

```
GOSUB solar_sensing
  I2cwrite DATAPIN,CLOCKPIN,ID2,dir_mem_2,[SUN]
  dir_mem_2=dir_mem_2+2
pause 20
```

```
i2cwrite DATAPIN,CLOCKPIN,ID2,dir_mem_2,[minutos]
dir_mem_2=dir_mem_2+1
pause 20
```

```
i2cwrite DATAPIN,CLOCKPIN,ID2,dir_mem_2,[hora]
```

```

dir_mem_2=dir_mem_2+1
pause 20

i2cwrite DATAPIN,CLOCKPIN,ID2,dir_mem_2,[dia_fecha]
dir_mem_2=dir_mem_2+1
pause 20

i2cwrite DATAPIN,CLOCKPIN,ID,dir_mem,[mes]
dir_mem_2=dir_mem_2+1
pause 20

SEROUT2 TX,84,[" DATOS GRABADOS ",13,10]
pause 500

adr_mem_2=adr_mem_2+1

endif

goto TOMA_DATO

'===== CICLO TRANSFERIR =====

TRANSFERIR:

IF TRANSFER==0 THEN TOMA_DATO

SEROUT2 TX,84,[" Ciclo de Transferir",13,10]

if adr_read<=4095 then

'===== LECTURA DE MEMORIA 1 =====

SEROUT2 TX,84,[" CICLO TRANSFERIR DATOS MEMORIA 1",13,10]
pause 500

SEROUT2 TX,84,[" LECTURA DE DATOS ",13,10]
pause 500

i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux1]
pause 20

SEROUT2 TX,84,[" VELOCIDAD ES = ",dec aux1," direccion ",dec dir_mem_aux,13,10]
pause 500
dir_mem_aux=dir_mem_aux+2

i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux2]
pause 20

SEROUT2 TX,84,[" SOL ES = ",DEC aux2," direccion ",dec dir_mem_aux,13,10]
pause 500
dir_mem_aux=dir_mem_aux+2

i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux3]
pause 20
dir_mem_aux=dir_mem_aux+1

i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux4]
pause 20
dir_mem_aux=dir_mem_aux+1

```

```

i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux5]
pause 20
dir_mem_aux=dir_mem_aux+1

i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux6]
pause 20

SEROUT2 TX,84,[" HORA: ",HEX2 aux4," : ",HEX2 aux3," FECHA: ",HEX2 aux5," MES: ",hex2 aux6, "
Direccion: ",dec dir_mem_aux,13,10]
pause 500
dir_mem_aux=dir_mem_aux+1

adr_read=adr_read+1

goto TRANSFERIR

'===== FIN LECTURA DE MEMORIA 1 =====

ELSE

'----- LECTURA DE MEMORIA 2 -----

SEROUT2 TX,84,[" CICLO TRANSFERIR DATOS MEMORIA 2",13,10]
pause 500

SEROUT2 TX,84,[" LECTURA DE DATOS ",13,10]
pause 500

i2cread DATAPIN,CLOCKPIN,ID2,dir_mem_aux2,[aux1]
pause 20

SEROUT2 TX,84,[" VELOCIDAD ES = ",dec aux1," direccion ",dec dir_mem_aux2,13,10]
pause 500
dir_mem_aux2=dir_mem_aux2+2

i2cread DATAPIN,CLOCKPIN,ID2,dir_mem_aux2,[aux2]
pause 20

SEROUT2 TX,84,[" SOL ES = ",DEC aux2," direccion ",dec dir_mem_aux2,13,10]
pause 500
dir_mem_aux2=dir_mem_aux2+2

i2cread DATAPIN,CLOCKPIN,ID2,dir_mem_aux2,[aux3]
pause 20
dir_mem_aux2=dir_mem_aux2+1

i2cread DATAPIN,CLOCKPIN,ID2,dir_mem_aux2,[aux4]
pause 20
dir_mem_aux2=dir_mem_aux2+1

i2cread DATAPIN,CLOCKPIN,ID2,dir_mem_aux2,[aux5]
pause 20
dir_mem_aux2=dir_mem_aux2+1

i2cread DATAPIN,CLOCKPIN,ID2,dir_mem_aux2,[aux6]
pause 20

SEROUT2 TX,84,[" HORA: ",HEX2 aux4," : ",HEX2 aux3," FECHA: ",HEX2 aux5," MES: ", hex2 aux6,"
Direccion: ",dec dir_mem_aux2,13,10]
pause 500

```

```
dir_mem_aux2=dir_mem_aux2+1
```

```
adr_read2 = adr_read2 + 1
```

```
GOTO TRANSFERIR
```

```
endif
```

```
'===== Indicador ====='
```

```
indicador:
```

```
  FOR i=1 to 3
```

```
    HIGH LEDV : PAUSE 500 : low LEDV : PAUSE 500
```

```
  NEXT i
```

```
return
```

```
'===== FIN Indicador ====='
```

```
'===== GRABAR RTC ====='
```

```
grabar_rtc:
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,0,[$00] '00 Segundos
```

```
  PAUSE 10
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,1,[$05] '00 Minutos
```

```
  PAUSE 10
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,2,[$14] '15 Horas
```

```
  PAUSE 10
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,3,[$04] 'Viernes (día de la semana)
```

```
  PAUSE 10
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,4,[$23] 'Día 18 del mes
```

```
  PAUSE 10
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,5,[$11] 'Noviembre = 11
```

```
  PAUSE 10
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,6,[$05] 'Año = 05
```

```
  PAUSE 10
```

```
  I2CWRITE DATAPIN,CLOCKPIN,%11010000,7,[$10] 'Encender el led cada 1 segundo
```

```
  PAUSE 10
```

```
return
```

```
'===== FIN GRABAR RTC ====='
```

```
'===== SPEED SENSING ====='
```

```
lectura_speed:
```

```
  COUNT VELOC,1000,speed
```

```
  pause 20
```

```
  Serout2 TX,84,["LA VELOCIDAD ES: ",dec speed,13,10]
```

```
return
```

```
'===== FIN SPEED SENSING ====='
```

```
'===== SOLAR SENSING ====='
```

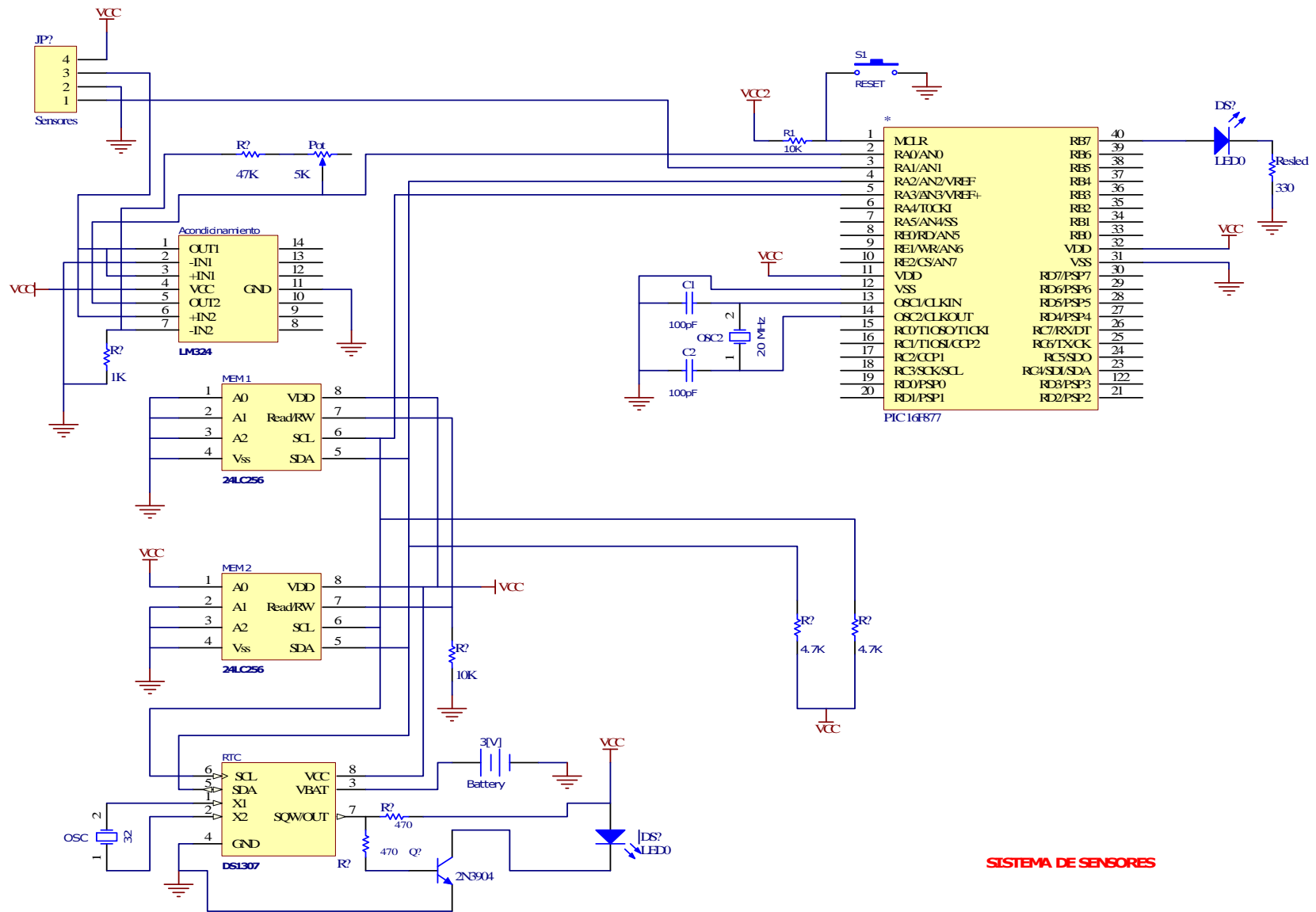
```
solar_sensing:
```

```
  ADCIN 0,SUN
```

```
  pause 20
```

```
  serout2 TX,84,["LA RADIACION SOLAR ES: ",dec SUN,13,10]
```


ANEXO F
DIAGRAMA DEL CIRCUITO DE SENSORES ACOPLADO
AL CIRCUITO DE ALMACENAMIENTO DE DATOS



ANEXO G
HOJA DE DATOS TÉCNICA DEL MICROCONTROLADOR 16C745



PIC16C745/765

8-Bit CMOS Microcontrollers with USB

Devices included in this data sheet:

- PIC16C745
- PIC16C765

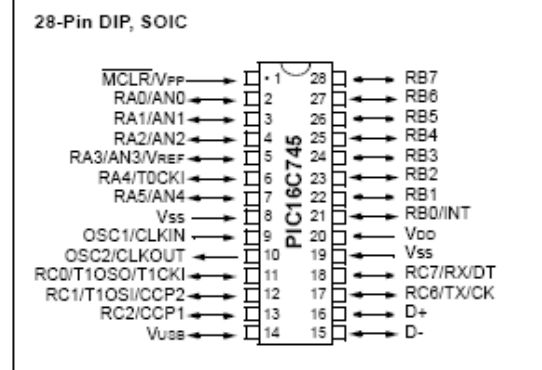
Microcontroller Core Features:

- High-performance RISC CPU
- Only 35 single word instructions

Device	Memory		Pins	A/D Resolution	A/D Channels
	Program x14	Data x8			
PIC16C745	8K	256	28	8	5
PIC16C765	8K	256	40	8	8

- All single cycle instructions except for program branches which are two cycle
- Interrupt capability (up to 12 internal/external interrupt sources)
- Eight level deep hardware stack
- Direct, indirect and relative addressing modes
- Power-on Reset (POR)
- Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Brown-out detection circuitry for Brown-out Reset (BOR)
- Programmable code-protection
- Power saving SLEEP mode
- Selectable oscillator options
 - EC - External clock (24 MHz)
 - E4 - External clock with PLL (6 MHz)
 - HS - Crystal/Resonator (24 MHz)
 - H4 - Crystal/Resonator with PLL (6 MHz)
- Processor clock of 24 MHz derived from 6 MHz crystal or resonator
- Fully static low-power, high-speed CMOS
- In-Circuit Serial Programming™ (ICSP)
- Operating voltage range
 - 4.35 to 5.25V
- High Sink/Source Current 25/25 mA
- Wide temperature range
 - Industrial (-40°C - 85°C)
- Low-power consumption:
 - ~ 16 mA @ 5V, 24 MHz
 - 100 µA typical standby current

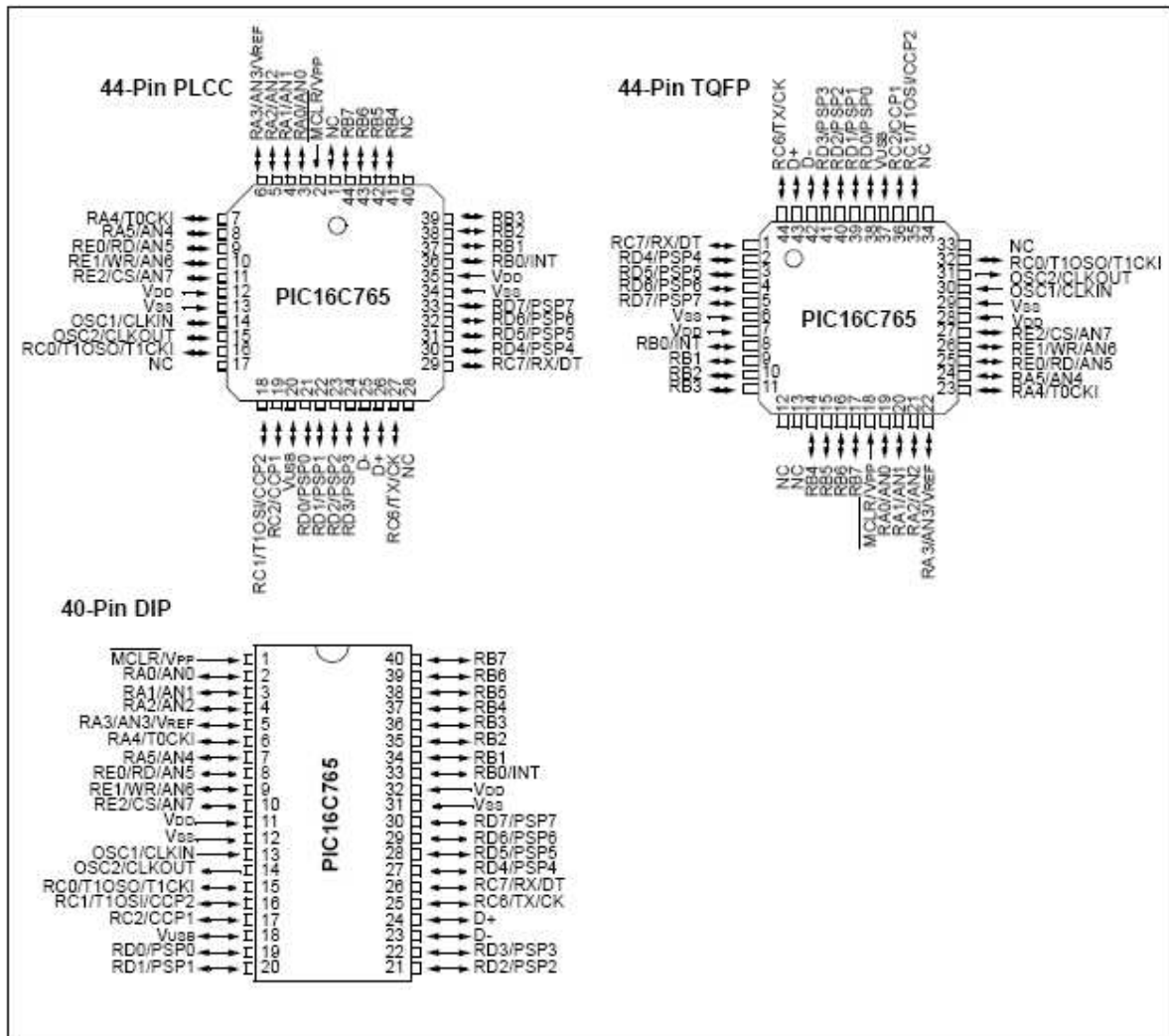
Pin Diagrams



Peripheral Features:

- Universal Serial Bus (USB 1.1)
 - Soft attach/detach
- 64 bytes of USB dual port RAM
- 22 (PIC16C745) or 33 (PIC16C765) I/O pins
 - Individual direction control
 - 1 high voltage open drain (RA4)
 - 8 PORTB pins with:
 - Interrupt-on-change control (RB<7:4> only)
 - Weak pull-up control
 - 3 pins dedicated to USB
- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler can be incremented during SLEEP via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- 2 Capture, Compare and PWM modules
 - Capture is 16-bit, max. resolution is 10.4 ns
 - Compare is 16-bit, max. resolution is 167 ns
 - PWM maximum resolution is 10-bit
- 8-bit multi-channel Analog-to-Digital converter
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI)
- Parallel Slave Port (PSP) 8-bits wide, with external \overline{RD} , \overline{WR} and \overline{CS} controls (PIC16C765 only)

PIC16C745/765



Key Features PICmicro™ Mid-Range Reference Manual (DS33023)	PIC16C745	PIC16C765
Operating Frequency	6 MHz or 24 MHz	6 MHz or 24 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Program Memory (14-bit words)	8K	8K
Data Memory (bytes)	256	256
Dual Port Ram	64	64
Interrupt Sources	11	12
I/O Ports	22 (Ports A, B, C)	33 (Ports A, B, C, D, E)
Timers	3	3
Capture/Compare/PWM modules	2	2
Analog-to-Digital Converter Module	5 channel x 8 bit	8 channel x 8 bit
Parallel Slave Port	—	Yes
Serial Communication	USB, USART/SCI	USB, USART/SCI
Brown-out Detect Reset	Yes	Yes

PIC16C745/765

TABLE 3-1: PIC16C745/765 PINOUT DESCRIPTION

Name	Function	Input Type	Output Type	Description
MCLR/VPP	MCLR	ST	—	Master Clear
	VPP	Power	—	Programming Voltage
OSC1/CLKIN	OSC1	Xtal	—	Crystal/Resonator
	CLKIN	ST	—	External Clock Input
OSC2/CLKOUT	OSC2	—	Xtal	Crystal/Resonator
	CLKOUT	—	CMOS	Internal Clock (FINT/4) Output
RA0/AN0	RA0	ST	CMOS	Bi-directional I/O
	AN0	AN	—	A/D Input
RA1/AN1	RA1	ST	CMOS	Bi-directional I/O
	AN1	AN	—	A/D Input
RA2/AN2	RA2	ST	CMOS	Bi-directional I/O
	AN2	AN	—	A/D Input
RA3/AN3/VREF	RA3	ST	CMOS	Bi-directional I/O
	AN3	AN	—	A/D Input
	VREF	AN	—	A/D Positive Reference
RA4/T0CKI	RA4	ST	OD	Bi-directional I/O
	T0CKI	ST	—	Timer 0 Clock Input
RA5/AN4	RA5	ST	—	Bi-directional I/O
	AN4	AN	—	A/D Input
RB0/INT	RB0	TTL	CMOS	Bi-directional I/O ⁽¹⁾
	INT	ST	—	Interrupt
RB1	RB1	TTL	CMOS	Bi-directional I/O ⁽¹⁾
RB2	RB2	TTL	CMOS	Bi-directional I/O ⁽¹⁾
RB3	RB3	TTL	CMOS	Bi-directional I/O ⁽¹⁾
RB4	RB4	TTL	CMOS	Bi-directional I/O with Interrupt-on-Change ⁽¹⁾
RB5	RB5	TTL	CMOS	Bi-directional I/O with Interrupt-on-Change ⁽¹⁾
RB6/ICSPC	RB6	TTL	CMOS	Bi-directional I/O with Interrupt-on-Change ⁽¹⁾
	ICSPC	ST	—	In-Circuit Serial Programming Clock Input
RB7/ICSPD	RB7	TTL	CMOS	Bi-directional I/O with Interrupt-on-Change ⁽¹⁾
	ICSPD	ST	CMOS	In-Circuit Serial Programming Data I/O
RC0/T1OSO/T1CKI	RC0	ST	CMOS	Bi-directional I/O
	T1OSO	—	Xtal	T1 Oscillator Output
	T1CKI	ST	—	T1 Clock Input
RC1/T1OSI/CCP2 ⁽¹⁾	RC1	ST	CMOS	Bi-directional I/O
	T1OSI	Xtal	—	T1 Oscillator Input
	CCP2	—	—	Capture In/Compare Out/PWM Out 2
RC2/CCP1	RC2	ST	CMOS	Bi-directional I/O
	CCP1	—	—	Capture In/Compare Out/PWM Out 1
VUSB	VUSB	—	Power	Regulator Output Voltage
D-	D-	USB	USB	USB Differential Bus
D+	D+	USB	USB	USB Differential Bus

Legend: OD = open drain, ST = Schmitt Trigger

Note 1: Weak pull-ups. PORT B pull-ups are byte wide programmable.

2: PIC16C765 only.

PIC16C745/765

TABLE 3-1: PIC16C745/765 PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RC6/TX/CK	RC6	ST	CMOS	Bi-directional I/O
	TX	—	CMOS	USART Async Transmit
	CK	ST	CMOS	USART Master Out/Slave In Clock
RC7/RX/DT	RC7	ST	CMOS	Bi-directional I/O
	RX	ST	—	USART Async Receive
	DT	ST	CMOS	USART Data I/O
RD0/PSP0	RD0	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP0	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RD1/PSP1	RD1	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP1	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RD2/PSP2	RD2	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP2	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RD3/PSP3	RD3	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP3	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RD4/PSP4	RD4	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP4	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RD5/PSP5	RD5	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP5	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RD6/PSP6	RD6	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP6	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RD7/PSP7	RD7	TTL	CMOS	Bi-directional I/O ⁽²⁾
	PSP7	TTL	—	Parallel Slave Port Data Input ⁽²⁾
RE0/ \overline{RD} /AN5	RE0	ST	CMOS	Bi-directional I/O ⁽²⁾
	\overline{RD}	TTL	—	Parallel Slave Port Control Input ⁽²⁾
	AN5	AN	—	A/D Input ⁽²⁾
RE1/ \overline{WR} /AN6	RE1	ST	CMOS	Bi-directional I/O ⁽²⁾
	\overline{WR}	TTL	—	Parallel Slave Port Control Input ⁽²⁾
	AN6	AN	—	A/D Input ⁽²⁾
RE2/ \overline{CS} /AN7	RE2	ST	CMOS	Bi-directional I/O ⁽²⁾
	\overline{CS}	TTL	—	Parallel Slave Port Data Input ⁽²⁾
	AN7	AN	—	A/D Input ⁽²⁾
VDD	VDD	Power	—	Power
VSS	VSS	Power	—	Ground

Legend: OD = open drain, ST = Schmitt Trigger

Note 1: Weak pull-ups. PORT B pull-ups are byte wide programmable.

2: PIC16C765 only.

PIC16C745/765

10.0 UNIVERSAL SERIAL BUS

10.1 Overview

This section introduces a minimum amount of information on USB. If you already have basic knowledge of USB, you can safely skip this section. If terms like Enumeration, Endpoint, IN/OUT Transactions, Transfers and Low Speed/Full Speed are foreign to you, read on.

USB was developed to address the increased connectivity needs of PC's in the PC 2000 specification. There was a base requirement to increase the bandwidth and number of devices, which could be attached. Also desired were the ability for hot swapping, user friendly operation, robust communications and low cost. The primary promoters of USB are Intel, Compaq, Microsoft and NEC.

USB is implemented as a Tiered Star topology, with the host at the top, hubs in the middle, spreading out to the individual devices at the end. USB is limited to 127 devices on the bus, and the tree cannot be more than 6 levels deep.

USB is a host centric architecture. The host is always the master. Devices are not allowed to "speak" unless "spoken to" by the host.

Transfers take place at one of two speeds. Full Speed is 12 Mb/s and Low Speed is 1.5 Mb/s. Full Speed covers the middle ground of data intensive audio and compressed video applications, while low speed supports less data intensive applications.

10.1.1 TRANSFER PROTOCOLS

Full speed supports four transfer types: Isochronous, Bulk, Interrupt and Control. Low speed supports two transfer types: Interrupt and Control. The four transfer types are described below.

- **Isochronous Transfers**, meaning equal time, guarantee a fixed amount of data at a fixed rate. This mode trades off guaranteed data accuracy for guaranteed timeliness. Data validity is not checked because there isn't time to re-send bad packets anyway and the consequences of bad data are not catastrophic.
- **Bulk Transfers** are the converse of Isochronous. Data accuracy is guaranteed, but timeliness is not.
- **Interrupt Transfers** are designed to communicate with devices which have a moderate data rate requirement. Human Interface Devices like keyboards are but one example. For Interrupt Transfers, the key is the desire to transfer data at regular intervals. USB periodically polls these devices at a fixed rate to see if there is data to transfer.
- **Control Transfers** are used for configuration purposes.

10.1.2 FRAMES

Information communicated on the bus is grouped in a format called Frames. Each Frame is 1 ms in duration and is composed of multiple transfers. Each transfer type can be repeated more than once within a frame.

10.1.3 POWER

Power has always been a concern with any device. With USB, 5 volt power is now available directly from the bus. Devices may be self-powered or bus-powered. Self-powered devices will draw power from a wall adapter or power brick. On the other hand, bus-powered devices will draw power directly from the USB bus itself. There are limits to how much power can be drawn from the USB bus. Power is expressed in terms of "unit loads" (≤ 100 mA). All devices, including Hubs, are guaranteed at least 1 unit load (low power), but must negotiate with the host for up to 5 unit loads (high power). If the host determines that the bus as currently configured cannot support a device's request for more unit loads, the device will be denied the extra unit loads and must remain in a low power configuration.

10.1.4 END POINTS

At the lowest level, each device controls one or more endpoints. An endpoint can be thought of as a virtual port. Endpoints are used to communicate with a device's functions. Each endpoint is a source or sink of data. Endpoints have both an In and Out direction associated with it. Each device must implement endpoint 0 to support Control Transfers for configuration. There are a maximum of 15 endpoints available for use by each full speed device and 6 endpoints for each slow speed device. Remember that the bus is host centric, so In/Out is with respect to the host and not the device.

10.1.5 ENUMERATION

Prior to communicating on the bus, the host must see that a new device has been connected and then go through an "enumeration process". This process allows the host to ask the device to introduce itself, and negotiate performance parameters, such as power consumption, transfer protocol and polling rate. The enumeration process is initiated by the host when it detects that a new device has attached itself to the bus. This takes place completely in the background from the application process.

10.1.6 DESCRIPTORS

The USB specification requires a number of different descriptors to provide information necessary to identify a device, specify its endpoints, and each endpoint's function. The five general categories of descriptors are Device, Configuration, Interface, End Point and String.

PIC16C745/765

The Device descriptor provides general information such as manufacturer, product number, serial number, USB device class the product falls under, and the number of different configurations supported. There can only be one Device descriptor for any given application.

The Configuration descriptor provides information on the power requirements of the device and how many different interfaces are supported when in this configuration. There may be more than one configuration for each device, (i.e., a high power device may also support a low power configuration).

The Interface descriptor details the number of endpoints used in this interface, as well as the class driver to use should the device support functions in more than just one device class. There can only be one Interface descriptor for each configuration.

The Endpoint descriptor details the actual registers for a given function. Information is stored about the transfer types supported, direction (In/Out), bandwidth requirements and polling interval. There may be more than one endpoint in a device, and endpoints may be shared between different interfaces.

Many of the four descriptors listed above will reference or index different String descriptors. String descriptors are used to provide vendor specific or application specific information. They may be optional and are encoded in "Unicode" format.

10.1.7 DEVICE CLASSES/CLASS DRIVERS

Operating systems provide drivers which group functions together by common device types called classes. Examples of device classes include, but are not limited to, storage, audio, communications and HID (Human Interface). Class drivers for a given application are referenced in both the Device descriptor and Interface descriptor. Most applications can find a Class Driver which supports the majority of their function/command needs. Vendors who have a requirement for specific commands which are not supported by any of the standard class drivers may provide a vendor specific ".inf" file or driver for extra support.

10.1.8 SUMMARY

While a complete USB overview is beyond the scope of this document, a few key concepts must be noted. Low speed communication is designed for devices, which in the past, used an interrupt to communicate with the host. In the USB scheme, devices do not directly interrupt the processor when they have data. Instead the host periodically polls each device to see if they have any data. This polling rate is negotiated between the device and host, giving the system a guaranteed latency.

For more details on USB, see the USB V1.1 spec, available from the USB website at www.usb.org.

10.2 Introduction

The PIC16C745/765 USB peripheral module supports Low Speed control and interrupt (IN and OUT) transfers only. The implementation supports 3 endpoint numbers (0, 1, 2) for a total of 6 endpoints.

The following terms are used in the description of the USB module:

- MCU - The core processor and corresponding firmware
- SIE - Serial Interface Engine: That part of the USB that performs functions such as CRC generation and clocking of the D+ and D- signals.
- USB - The USB module including SIE and registers
- Bit Stuffing - forces insertion of a transition on D+ and D- to maintain clock synchronization
- BD - Buffer Descriptor
- BDT - Buffer Descriptor Table
- EP - Endpoint (combination of endpoint number and direction)
- IN - Packet transfer into the host
- OUT - Packet transfer out of the host

10.3 USB Transaction

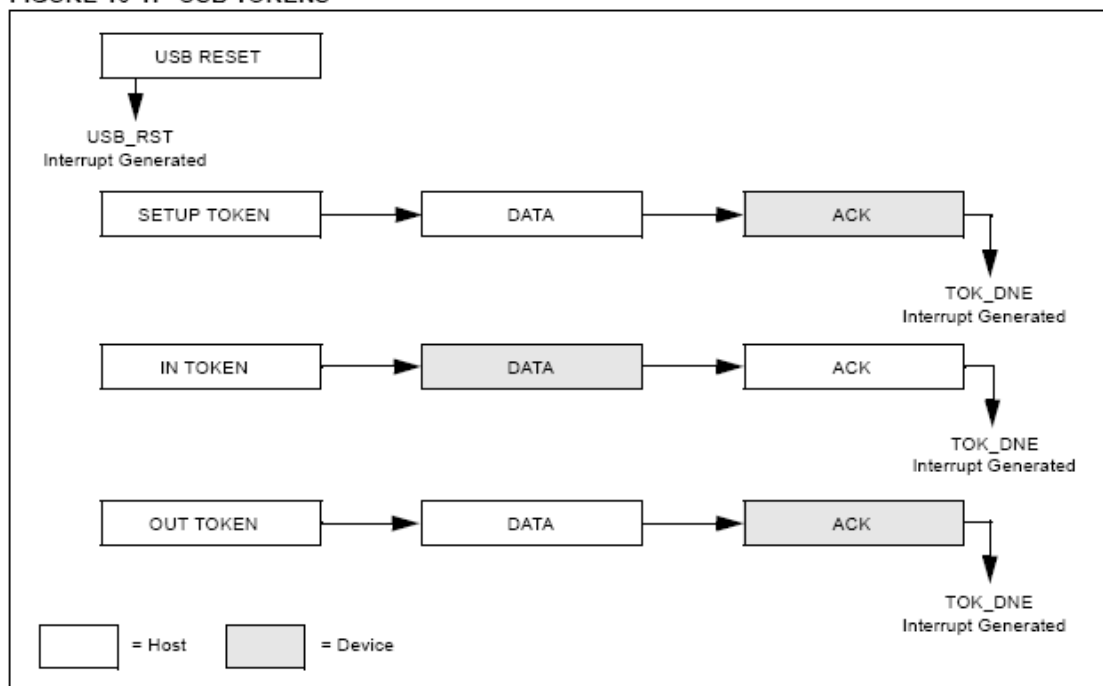
When the USB transmits or receives data the SIE will first check that the corresponding endpoint and direction Buffer Description UOWN bit equals 1. The USB will move the data to or from the corresponding buffer. When the TOKEN is complete, the USB will update the BD status and change the UOWN bit to 0. The USTAT register is updated and the TOK_DNE interrupt is set. When the MCU processes the TOK_DNE interrupt it reads the USTAT register, which gives the MCU the information it needs to process the endpoint. At this point the MCU will process the data and set the corresponding UOWN bit. Figure 10-1 shows a time line of how a typical USB token would be processed.

10.4 Firmware Support

Microchip provides a comprehensive support library of standard chapter 9 USB commands. These libraries provide a software layer to insulate the application software from having to handle the complexities of the USB protocol. A simple Put/Get interface is implemented to allow most of the USB processing to take place in the background within the USB interrupt service routine. Applications are encouraged to use the provided libraries during both enumeration and configured operation.

PIC16C745/765

FIGURE 10-1: USB TOKENS



ANEXO H
USB FIRMWARE DE MICROCHIP

USB Firmware Users Guide – Assembly version

V1.21 5 August 2001

1. Introducing the USB software interface

Microchip provides a layer of software for the PIC16C745/65 that handles the lowest level interface so your application won't have to. This provides a simple Put/Get interface for communication. Most of the USB processing takes place in the background through the Interrupt Service Routine. From the application viewpoint, the enumeration process and data communication takes place without further interaction. However substantial setup is required in the form of generating appropriate descriptors.

2. Integrating USB into your application

The latest version of the USB interface software is available on Microchip's website. See <http://www.microchip.com/>

The interface to the application is packaged in 5 functions: **InitUSB**, **PutEP1**, **PutEP2**, **GetEP1** and **GetEP2**. **InitUSB** initializes the USB peripheral allowing the host to enumerate the device. Then for normal data communications, the **PutEPn** functions send data to the host and the **GetEPn** functions receive data from the host.

There's also a fair amount of setup work that must be completed. USB depends heavily on the descriptors. These are the software parameters that are communicated to the host to let it know what the device is, and how to communicate with it. See USB V1.1 spec section 9.5 for more details.

InitUSB enables the USB interrupt so enumeration can begin. The actual enumeration process occurs in the background, driven by the host and interrupt service routine. Macro **ConfiguredUSB** waits until the device is in the CONFIGURED state. The time required to enumerate is completely dependent on the host and bus loading.

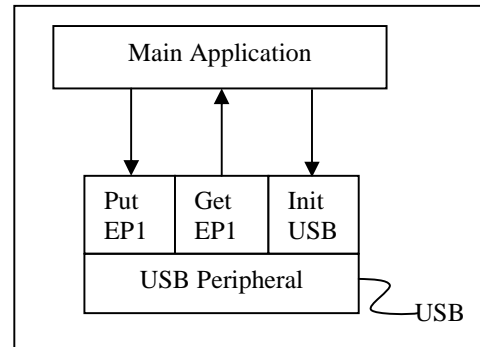
3. Interrupt structure concerns

Processor Resources

Most of the USB processing occurs via the interrupt and thus is invisible to the application. However it still consumes processor resources. These include ROM, RAM, Common RAM, Stack Levels and processor cycles. This section attempts to quantify the impact on each of these resources, and shows ways to avoid conflicts.

These considerations should be taken into account if you write your own Interrupt Service Routine: Save W, STATUS, FSR and PCLATH which are the file registers that may be corrupted by servicing the USB interrupt.

The file *usb_main.asm* provides a skeleton ISR which does this for you, and includes tests for each of the possible ISR bits. This provides a good starting point if you haven't already written your own.



Stack Levels

The hardware stack on the PIC is only 8 levels deep. So the worst case call between the application and ISR can only be 8 levels. The enumeration process requires 4 levels, so it's best if the main application holds off on any processing until enumeration is complete. **ConfiguredUSB** is a macro that waits until the enumeration process is complete for exactly this purpose. This macro does this by testing the lower two bits of USWSTAT (0x197).

ROM

The code required to support the USB interrupt, including the chapter 9 interface calls, but not including the HID interface calls or descriptor tables is about 1kW. The HID specific functions and descriptor tables take up additional memory. The location of these parts is not restricted, and the linker script may be edited to control the placement of each part. See the Strings and Descriptors sections in the linker script.

RAM

With the exception of Common RAM discussed below, servicing the USB interrupt requires approximately 40 bytes of RAM in Bank 2. That leaves all the General Purpose RAM in banks zero and one, plus half of Bank 2 available for your application to use.

Common RAM usage

The PIC16C745/765 has 16 bytes of common RAM. These are the last 16 addresses in each bank and all refer to the same 16 bytes of memory without regard to which register bank is currently addressed by the RP0, RP1 and IRP bits.

These are particularly useful when responding to interrupts. When an interrupt occurs, the ISR doesn't immediately know which bank is addressed. With devices that don't support common RAM, the W register must be provided for in each bank. The PIC16C745/765 can save the appropriate registers in Common RAM and not have to waste a byte in each bank for the W register.

Buffer allocation

The PIC16C745/765 has 64 bytes of Dual Port RAM. 24 are used for the Buffer Descriptor Table, (BDT) leaving 40 bytes for buffers.

Endpoint 0 (EP0) IN and OUT need dedicated buffers since a setup transaction can never be NAKed. That leaves three buffers for four possible endpoints. But the USB spec requires that low speed devices are only allowed 2 endpoints (USB V1.1 paragraph 5.3.1.2), where an endpoint is a simplex connection that is defined by the combination of endpoint number and direction.

The default configuration allocates individual buffers to EP0 OUT, EP0 IN, EP1 OUT, and EP1 IN. The last buffer is shared between EP2 IN and EP2 OUT. Again, the spec says low speed devices can only use 2 endpoints beyond EP0. This configuration supports most of the possible combinations of endpoints (EP1 OUT and EP1 IN, EP1 OUT and EP2 IN, EP1 OUT and EP2 OUT, EP1 IN and EP2 OUT, EP1 IN and EP2 IN). The only combination that is not supported by this configuration is EP2 IN and EP2 OUT. If your application needs both EP2 IN and EP2 OUT, the function **USBReset** will need to be edited to give each of these endpoints dedicated buffers at the expense of EP1.

4. File packaging

The software interface is packaged into five files, designed to simplify the integration with your application. These are:

1. usb_main.asm
2. usb_ch9.asm
3. hidclass.asm
4. descript.asm
5. usb_defs.inc

File *usb_main.asm* is useful as a starting point on a new application and as an example of how an existing application needs to service the USB interrupt and communicate with the core functions.

File *usb_ch9.asm* contains the interface and core functions needed to enumerate the bus. It also contains the functions that service the USB, send data to the host, and receive data from the host.

File *hidclass.asm* provides some HID Class specific functions. Currently, only **Get_Report_Descriptor** and **Set_Report** are supported. Other class specific functions can be implemented in a similar fashion. When a get token interrupt determines that it is a class specific command on the basis that ReportType bit 6 is set, control is passed to function **ClassSpecific**. If you're working with a different class, this is your interface between the core functions and the class specific functions.

File *descript.asm* contains the device, config, interface, endpoint and string descriptors. It also includes the HID specific descriptors -- HID and Report.

File *usb_defs.inc* contains several macros including **ConfigUSB**, **PutEP1**, **PutEP2**, **GetEP1**, and **GetEP2**. As mentioned before, instances of the put and get macros have been created in *usb_ch9.asm* already.

5. Function Call reference

Interface between the Application and Protocol layer takes place in five main functions: **InitUSB**, **PutEP1**, **PutEP2**, **GetEP1** and **GetEP2**.

InitUSB should be called by the main program after the device is powered up. Be sure to precede the call with a 16 μ s delay to give the SIE a chance to reset before beginning enumeration. **InitUSB** enables the USB peripheral and the USB reset interrupt. It also transitions the part to the powered state in order to prepare the device for enumeration. Before enumeration is initiated, the USB Reset is the only USB interrupt allowed, preventing the part from responding to anything on the bus until it's been reset. The USB Reset interrupt initializes the Buffer Descriptor Table (BDT) and transitions the part to the default state where it responds to commands on address zero. When it receives a SET ADDRESS command, the device transitions to the addressed state and now responds to commands on the new address.

PutEPn sends data to the host. Call this function with the following specified:

- The beginning address of the block of data to transmit in FSR/IRP.
- The number of bytes in the block of data in W.

If the IN buffer is available for that endpoint, the block of data is copied to the buffer, then the Data 0/1 bit is flipped and the owns bit is set. Should the buffer not be available, a failure code will be returned so the application can try again later (the Carry Flag will be cleared.) If the function is successful it will set the Carry Flag.

GetEPn returns data sent from the host. Call this function with the beginning address of the desired location data will be placed in FSR/IRP. If there is a buffer ready, meaning data has been received from the host, it is copied to the destination pointed to by FSR/IRP. The function will return the number of bytes copied in the W register. If no data is available, it returns a failure code (the Carry Flag is cleared.) If the function is successful it will set the Carry Flag.

ServiceUSBInt handles all interrupts generated by the USB peripheral. First it copies the active buffer to common RAM which provides a quick turn around on the buffer in dual port RAM and also to avoids having to switch banks during processing of the buffer.

StallUSBEP/UnstallUSBEP sets or clears the stall bit in the endpoint control register. The stall bit indicates to the host that user intervention is required and until such intervention is made, further attempts to communicate with the endpoint will not be successful. Once the user intervention has been made, **UnstallUSBEP** will clear the bit allowing communications to take place. These calls are used to signal to the host that user intervention is required. An example of this might be a printer out of paper.

SoftDetachUSB clears the DEV_ATT bit, electrically disconnecting the device from the bus, then reconnecting so it can be re-enumerated by the host. This process takes approximately 50 mS, to ensure that the host has seen the device disconnect and reattach to the bus.

CheckSleep tests the UCTRL.IDLE bit which if set, indicates that there has been no activity on the bus for 3 mS. If set, the device can be put to sleep, which puts the part into a low power standby mode until awakened by bus activity. This has to be handled outside the ISR because we need the interrupt to wake the PICmicro from sleep, and also because the application may not be ready to sleep when the interrupt occurs. Instead, the application should periodically call this function in order to poll the bit and find out when the device is in a good place to sleep.

Prior to putting the device to sleep, it enables the activity interrupt so the device will be awakened by the first transition on the bus. The PIC will immediately jump to the ISR which recognizes the activity interrupt and then disables the interrupt and resumes processing with the instruction following the **CheckSleep** call.

ConfiguredUSB (a macro) continuously polls the enumeration status bits and waits until the device has been configured by the host. This should be used after the call to **InitUSB** and prior to the first time your application attempts to communicate on the bus.

SetConfiguration is a callback function that allows your application to associate some meaning to a Set Configuration command from the host. The Ch9 software stores the value in USB_Curr_Config so it can be reported back on a **Get_Configuration** call. This function is also called, passing the new configuration into W. This function is called from within the ISR so it should be kept as short as possible.

6. Behind the scenes:

InitUSB clears the error counters and enables the 3.3 V regulator and the USB reset interrupt. This implements the requirement to prevent the PIC from responding to commands until the device has been reset.

The host sees the device, and resets the device to begin the enumeration process. The reset then initializes the Buffer Descriptor Table (BDT) and EndPoint Control Registers in addition to enabling the remaining USB interrupt sources.

The Interrupt transfers control to the interrupt vector (address 0x0004). Any interrupt service routine must preserve the processor state by saving the FSRs that might change during interrupt processing. We recommend saving W, STATUS, PCLATH and FSR. W can be stored in Common RAM to avoid banking issues. Then it starts polling the Interrupt flags to see what triggered the interrupt. The USB interrupts are serviced by calling **ServiceUSBInt** which further tests the USB interrupt sources to determine how to process the interrupt.

Then the host sends a setup token requesting the device descriptor. The USB Peripheral receives the Setup transaction, places the data portion in the EP0 out buffer, loads the USTAT register to indicate which endpoint received the data and triggers the Token Done (TOK_DNE) interrupt. The Chapter 9 commands then interpret the Setup token and sets up the data to respond to the request in the EP0 IN buffer. It then sets the UOWN bit to tell the SIE there is data available.

Next, the host sends an IN transaction to receive the data from the setup transaction. The SIE sends the data from the EP0 IN buffer and then sets the Token done interrupt to notify us that the data has been sent. If there is additional data, the next buffer full is set up in the EP0 IN buffer.

This token processing sequence holds true for the entire enumeration sequence, which walks through the flow chart starting chapter 9 of the USB spec. The device starts off in the powered state, transitions to RESET via the reset interrupt, transitions to ADDRESSED via the Set Address command, and transitions to CONFIGURED via a Set Configuration command.

The USB peripheral detects several different errors and handles most internally. The USB_ERR interrupt notifies the PIC that an error has occurred. No action is required by the PIC when an error occurs. Instead the errors are simply acknowledged and counted. There is no mechanism to pull the device off the bus if there are too many errors. If this behavior is desired it must be implemented in the application.

The Activity interrupt is left disabled until the USB peripheral detects no bus activity for 3 mS. Then it suspends the USB peripheral and enables the activity interrupt. The activity interrupt then reactivates the USB peripheral when bus activity resumes so processing may continue.

CheckSleep is a separate call that takes the bus idle one step further and puts the PIC to sleep if the USB peripheral has detected no activity on the bus. This powers down most of the device to minimal current draw. This call should be made at a point in the main loop where all other processing is complete.

7. Examples

This example shows how the USB functions are used. This example first initializes the USB peripheral which allows the host to enumerate the device. The enumeration process occurs in the background, via an Interrupt service routine. This function waits until enumeration is complete, and then polls EP1 OUT to see if there is any data available. When a buffer is available, it is copied to the IN buffer. Presumably your application would do something more interesting with the data than this example.

```
; *****
; Demo program that initializes the USB peripheral, allows the Host
; to Enumerate, then copies buffers from EP1OUT to EP1IN.
; *****
Main
    pagesel    InitUSB
    call       InitUSB        ; Set up everything so we can enumerate
    pagesel    Main
    ConfiguredUSB            ; wait here until we have enumerated.

CheckEP1        ; Check Endpoint 1 for an OUT transaction
    bankisel   BUFFER        ; point to lower banks
    pagesel   GetEP1
    movlw     BUFFER
    movwf    FSR              ; point FSR to our buffer
    call     GetEP1          ; If data is ready, it will be copied.
    pagesel   CheckEP1
    btfss    STATUS,C        ; was there any data for us?
    goto     CheckEP1        ; Nope, check again.

PutBuffer
    bankisel   BUFFER        ; point to lower banks
    pagesel   PutEP1
    movlw     BUFFER
    movwf    FSR              ; point FSR to our buffer
    movlw    0x08            ; send 8 bytes to endpoint 1
    call     PutEP1
    pagesel   PutBuffer
    btfss    STATUS,C        ; was it successful?
    goto     PutBuffer        ; No: try again until successful
    pagesel   CheckEP1
    goto     CheckEP1        ; Yes: restart loop
end
```

8. Multiple Configuration or Report Descriptors

The Ch9 firmware makes allowances for the fact that more than one configuration or report descriptor may be desired. Allowances for multiple interface, HID, and endpoint descriptors were not made because they are not needed. These descriptors are all read in with the configuration descriptor regardless of how many there are.

The host requests the descriptors by specifying the type of descriptor it wants and an index value. If more than one configuration descriptor exists it will request the first one by specifying an index of zero and the second by specifying an index of one. To make this process as easy as possible for developers to deal with, the functions **Config_descr_index** and **Report_descr_index** have been created in *descript.asm*. These functions will need to be modified in your code has more than one configuration or more than one report descriptor. All that is needed is for you to specify the starting label for additional descriptors in the lookup table for these functions.

*Note: String descriptors also use an index function, `string_index` (in *descript.asm*). This function will need to be modified in the same manner if your code has a number of strings other than six.*

9. Optimizing the Firmware

This firmware has been created to provide developers with ready-made USB functions so they don't have to create these functions for themselves. Most developers will not utilize all of the functions in the Ch9 firmware. In order to optimize the program memory, unused functions can be taken out of the firmware. The following guidelines are a good place to start the optimization.

USB status on Port B.

The firmware outputs the status of USB communication on Port B. This feature is intended for use with the PICDEM USB circuit board which drives an LED with each Port B pin. The LEDs indicate the following USB status information: RB0 – powered, RB1 – default, RB2 – addressed, RB3 – configured, RB4 – sleeping, RB5 – EP0 active, RB6 – EP1 active, RB7 – EP2 active. Obviously, these USB status indicators will probably not be used in a finished product by a developer although they are very helpful during development. All code associated with the USB status LEDs can be eliminated from the program memory by ensuring that `SHOW_ENUM_STATUS` is not defined at the top of *usb_ch9.asm*.

Error counter

Similar to the USB status LEDs, code exists in the firmware that counts various errors for debugging purposes. To eliminate this excess code from the program memory simply make sure that `COUNTERRORS` is not defined.

GetEP1, GetEP2, PutEP1, and PutEP2

These functions are all macros defined in *usb_defs.inc*. Instances of each of these macros occur in *usb_ch9.asm*. If a developer does not utilize one or more of these functions, space can be saved by removing the instance(s) not needed from *usb_ch9.asm*.

HID class

The HID class is one of several classes suitable for low-speed USB. In addition to these other classes, a vendor-defined class can be specified. Should a developer use a class other than the HID class, any HID class specific code in the firmware would be wasting space. The HID class specific code is found in the file, *hidclass.asm*. In a case where the HID class is not being utilized by a developer, this file and any variables or labels associated with it should be removed from the project.

10. Cursor Demonstration

Microchip provides a working USB demonstration. This demonstration has the effect of moving the cursor in a small circle on the user's screen. The following steps will get the demonstration working with an actual part or the MPLAB ICE:

Getting the USB demonstration to work on a PIC16C745/65

1. Unzip **usbxxxasm.zip** to a project folder.
2. Build the project in MPLAB.
3. Program a PIC16C765 using a PICSTART Plus or a PRO MATE II. Make sure the configuration bits are set as follows:

- Oscillator: H4
- Watchdog Timer: Off
- Power Up Timer: Off
- Code Protect: Off

Note: To program a PIC16C745 you will need to use the linker file and include files associated with the PIC16C745. Also, you will need to identify the part in the Development Mode dialog. See the MPLAB IDE User's Guide for details.

4. The figure at the bottom of this section details the circuit in which to implement the PIC. Attach the USB cable to your computer.
5. Provide power to the PIC. If you are running Windows 98(with the USB upgrade), Windows NT, or Windows 2000 then your operating system will detect a new device and install the necessary drivers automatically. After this occurs you should see the cursor rotating in a small circle on your screen. To stop the cursor from rotating, detach the USB cable.

Getting the USB demonstration to work using the MPLAB ICE

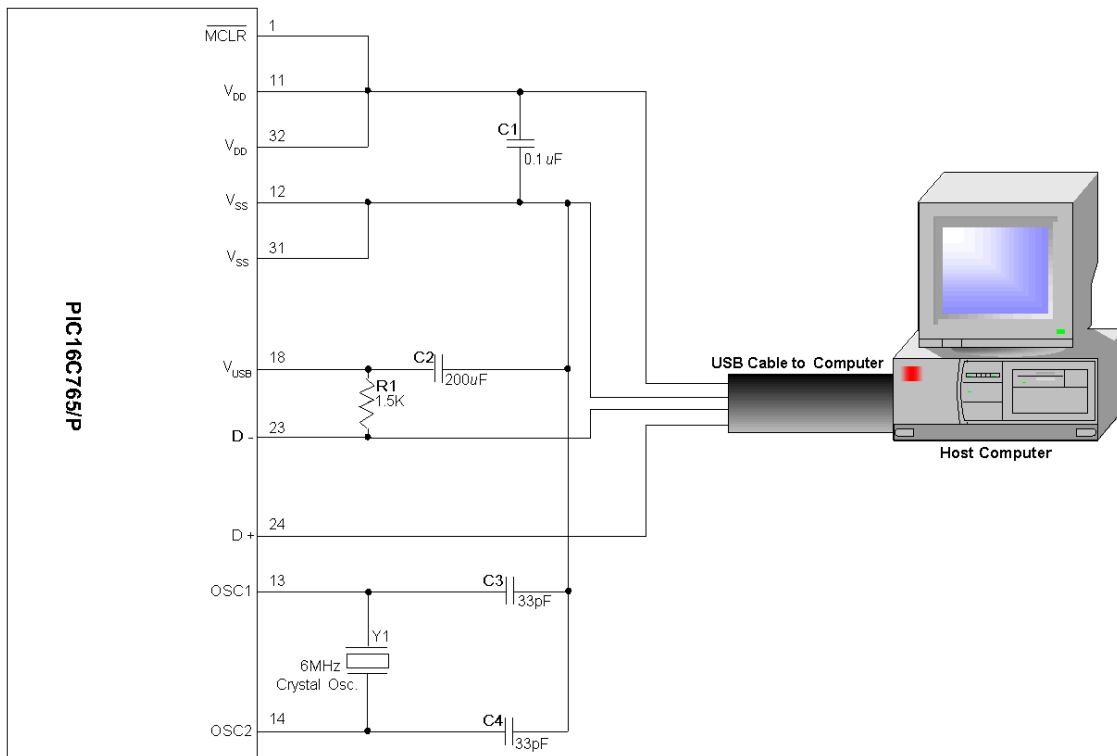
1. Unzip **usbxxxasm.zip** to a project folder.
2. Build the project in MPLAB.
3. Make sure the emulator is set up as follows in the Development Mode dialog:
 - Tools: MPLAB-ICE Emulator
 - Clock: Desired Frequency: 24 MHz
 - Configuration: Watch Dog Timer: None
 - Power: Processor Power: From Target Board

Note: The firmware is set up to run on a PIC15C765 as the default. To emulate a PIC16C745 you will need to use the linker file and include files associated with the PIC16C745. Also you will need to identify the part in the Development Mode dialog.

4. The figure below shows the circuit in which to implement the ICE. Connect the USB cable to your computer. (The external oscillator portion of the circuit is optional when using MPLAB ICE.)
5. Run the project on the emulator. If you are running Windows 98(with the USB upgrade), Windows NT, or Windows 2000 then your operating system will detect a new device and install the necessary drivers automatically. After this occurs you should see the cursor rotating in a small circle on your screen. To stop the cursor from rotating, press F5.

Emulation Tips

1. Turn on the emulator, then open the project to initialize the emulator. If the project is in a mode other than Emulation follow step 2.
2. If your project is already open before you turn the emulator on you can initialize it by first, turning the emulator on. Then go to the option menu and click Development Mode. Make sure that the MPLAB-ICE Emulator radio button is selected in the Tools folder of the Development Mode dialog. Click OK. The emulator will initialize.
3. Make sure the proper processor module and device adapter are installed.
4. Should you have other problems please refer to the MPLAB-ICE Users Guide as your first resource.



Note: Be sure to pull D- up to V_{USB} (via R1) *not* V_{DD} . For more on why this is done see the section **Universal Serial Bus:Transceiver:Regulator** in the PIC16C745/65 datasheet.

ANEXO I
CÓDIGO DEL PROGRAMA PARA LA COMUNICACIÓN DE DATOS USB

```

*****
* Name   : SISTEMA USB.BAS                               *
* Author : ANDRES VALLEJO y JUAN SERRANO                 *
* Notice : Copyright (c) 2005 []                          *
*       : All Rights Reserved                            *
* Date   : 23/11/2005                                    *
* Version : 1.1                                          *
* Notes  : COMUNICACION USB                              *
*       :                                                *
*****

```

'Codigo propio del Firmware USB

```

wsave var    byte $70 system
ssave var    byte bank0 system
psave var    byte bank0 system
fsave var    byte bank0 system

```

```

buffer var   byte[8] 'Buffer es una cadena de 8 bytes en la cual se envian los datos
cnt        var   byte 'Es el número de Bytes que se reciben (móximo 8)

```

'===== CONFIGURACION MEMORIA I2C =====

DEFINE I2C_HOLD 1

```

ID      CON %10100000 'ID de la memoria 1 I2C
ID2     CON %10100010 'ID de la memoria 2 I2C

```

'Declaracion de variables auxiliares

```

dir_mem_aux var word
dir_mem_aux=0
dir_mem_aux2 var word
adr_read    var word
adr_read2   var word
aux1        var byte
aux2        var byte

```

'===== FIN CONFIGURACION MEMORIA I2C =====

'===== CONFIGURACION I2C y PUERTOS =====

```

define I2C_SCLOUT 1 'Instrucción para evitar la necesidad de colocar resis-
                    'tencia pull-up en SCL
ADCON1 = 7         'Configura puerto A como digital
CLOCKPIN VAR PORTA.2 'Pin para señal de clock I2C
DATAPIN VAR PORTA.3 'Pin para señal de datos I2C
TRISB.7=0         'Seteo del puerto B7 como salida
TRISB.6=0         'Seteo del puerto B6 como salida
TRISB.5=0         'Seteo del puerto B5 como salida

```

'===== FIN DE CONFIGURACION I2C Y PUERTOS =====

'===== ASIGNO ALIAS A LOS PUERTOS =====

```

LED var PORTA.1 'Red/green LED on PORTA bits 2 and 5

```

```

LEDG var PORTA.5
MEM1 VAR PORTB.7 'Port B7 Asigno como MEM1
MEM2 VAR PORTB.6 'Port B6 Asigno como MEM2
PIC VAR PORTB.5 'Port B5 Asigno como PIC

'===== FIN ASIGNO ALIAS A LOS PUERTOS =====

'===== CONFIGURACION OSCILADOR =====

Define OSC 24 'Oscilador de 6 MHz (HS 4PLL )

'===== FIN CONFIGURACION OSCILADOR =====

' Uncomment following lines to display ENUM status on PORTB
'Define SHOW_ENUM_STATUS 1
' PORTB = 0 ' All LEDs off
' TRISB = 0 ' PORTB to all output

Goto start ' Skip around interrupt handler

' Define interrupt handler
define INTHAND usbint

' Assembly language interrupt handler
asm

; W, STATUS and PCLATH registers already saved, save FSR
usbint movf FSR, W
movwf fsave

; Check interrupt source and vector there
movlw high ServiceUSBInt
movwf PCLATH
btfsc PIR1, USBIF
call ServiceUSBInt

; Restore saved registers
clrf STATUS
movf fsave, W
movwf FSR
movf psave, W
movwf PCLATH
swapf ssave, W
movwf STATUS
swapf wsave, F
swapf wsave, W

retfie ; Return from interrupt

endasm

start:

Low LEDG ' Ground one end of LED

USBInit ' Init USB and wait till configured

```

```

        High LED           ' Turn on LED for USB ready
HIGH PIC
PAUSE 1000
HIGH MEM1
HIGH MEM2

'===== Espera recibir datos para comenzar =====

idleloop:

        USBIn 1, buffer, cnt, idleloop

'===== Datos Recibidos =====

bumpit:

'===== Lectura de las memorias I2C =====

        i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux1]
        pause 20
        buffer[0] = aux1
        dir_mem_aux=dir_mem_aux+1

        i2cread DATAPIN,CLOCKPIN,ID,dir_mem_aux,[aux2]
        pause 20
        buffer[1] = aux2
        dir_mem_aux=dir_mem_aux+1

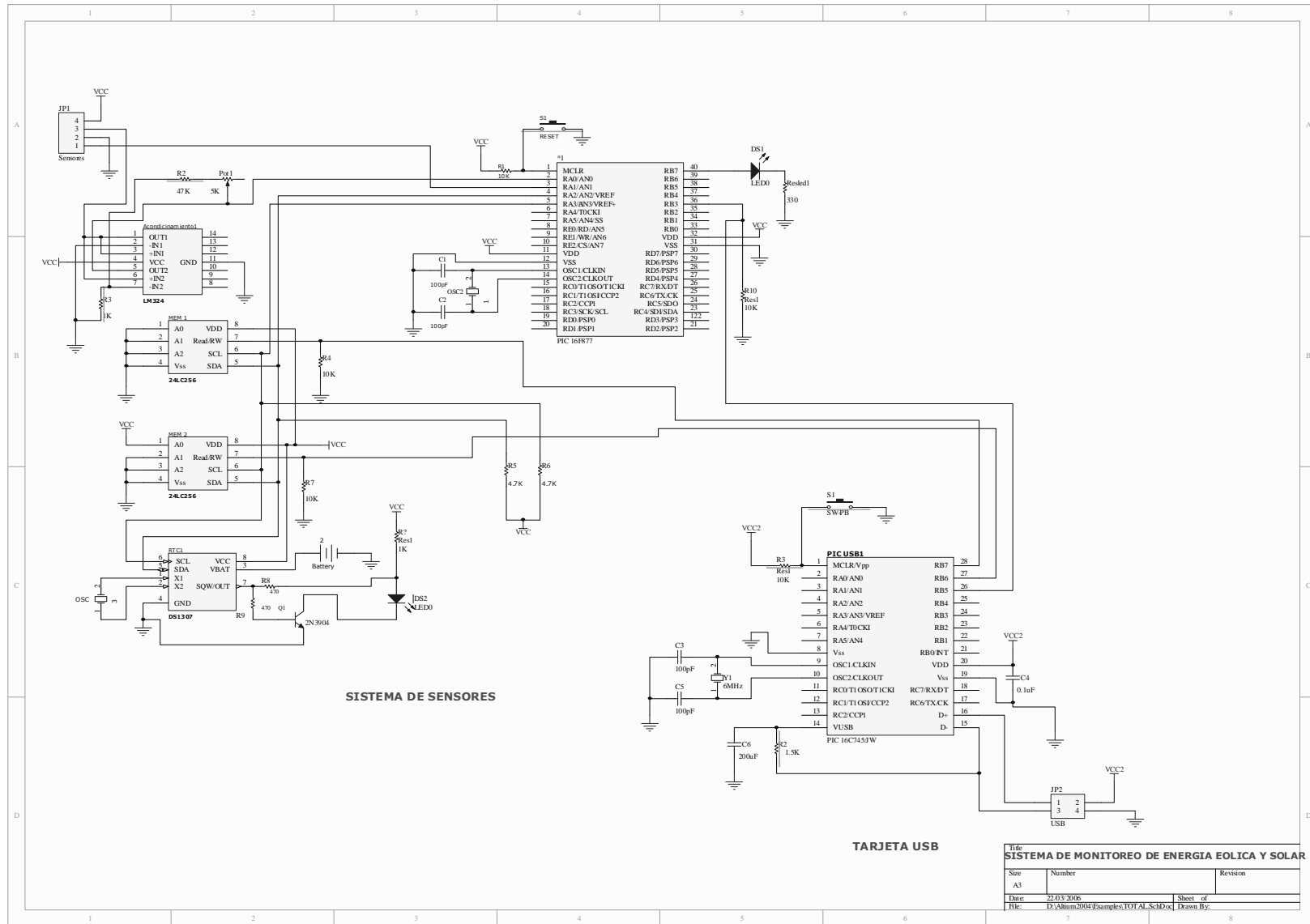
'===== Envio de Datos hacia el Puerto USB =====

        USBOut 1, buffer, 2, bumpit    ' Send the bytes back

Goto idleloop    ' Wait for next buffer

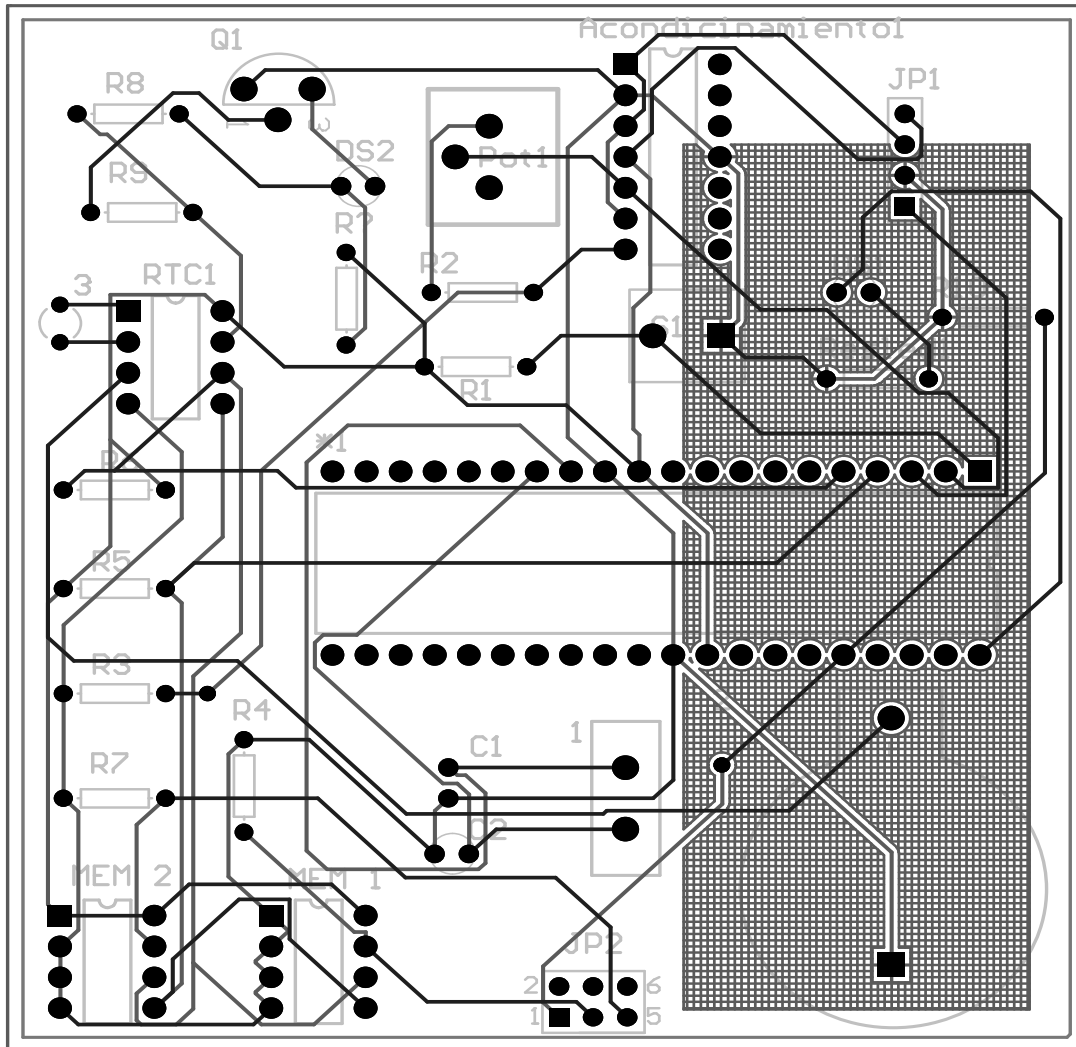
```


ANEXO J
DIAGRAMA DEL CIRCUITO DE MONITOREO
DE ENERGÍA EÓLICA Y SOLAR CON INTERFAZ USB



Sistema de Monitoreo de Energía Eólica y Solar con interfaz USB

ANEXO K
DIAGRAMA DE LAS PLACAS DEL CIRCUITO DE MONITOREO
DE ENERGÍA EÓLICA Y SOLAR CON INTERFAZ USB



ANEXO L
CODIGO DEL PROGRAMA DE LA INTERFAZ HOMBRE – MAQUINA (HMI)

FORMULARIOS
FORM 1

```

Private wks As Workspace
Private db As Database
Private rs_DatosR As Recordset
Dim cadena As String
Dim Buffer() As Byte
Dim BufferSize As Long
Dim i As Variant
Dim aux_orden As Integer
Dim NUM_REG As Integer
Dim num_rec As Integer
Private Sub Command1_Click()
'Dim Buffer() As Byte
'Dim BufferSize As Long
'Dim I As Variant
I = 0
Timer1.Interval = 500
    ReDim Buffer(16)
    HIDComm1.Connect
    Buffer(0) = CByte(Text1.Text)
    Buffer(1) = CByte(Text2.Text)
    BufferSize = 2
    HIDComm1.WriteTo Buffer, BufferSize
    Buffer = HIDComm1.ReadFrom(BufferSize)
    If BufferSize < 2 Then Exit Sub
    Text5(0).Text = Buffer(0)
    Text5(1).Text = Buffer(1)
    'Text5(2).Text = Buffer(2)
    'Text5(3).Text = Buffer(3)
    'Text5(4).Text = Buffer(4)
    'Text5(5).Text = Buffer(5)
    'Text5(6).Text = Buffer(6)
    'Text5(7).Text = Buffer(7)
End Sub
Private Sub Command2_Click()
Text3.Text = trans_Binario(Trim(Text4.Text)) + Text3.Text
End Sub
Private Sub Form_Activate()
HIDComm1.Connect
End Sub
Private Sub Form_Load()
aux_orden = 1
num_rec = 1
OpenDatabase
cadena = "select * from datos"
Set rs_DatosR = db.OpenRecordset(cadena)
Timer1.Interval = 10
End Sub
Private Sub Form_Terminate()
    HIDComm1.Uninit
End Sub
Private Sub rad()
Dim dat1 As String
Dim dat2 As String

```

```
Dim total As String
dat1 = Trim(Label1(0).Caption)
dat2 = Trim(Label4.Caption)
End Sub
Private Function trans_Binario(num As String) As String
Dim aux As String
Select Case num ' Evalúa Número.
Case "0"
    aux = "0000"
Case "1"
    aux = "0001"
Case "2"
    aux = "0010"
Case "3"
    aux = "0011"
Case "4"
    aux = "0100"
Case "5"
    aux = "0101"
Case "6"
    aux = "0110"
Case "7"
    aux = "0111"
Case "8"
    aux = "1000"
Case "9"
    aux = "1001"
Case "A"
    aux = "1010"
Case "B"
    aux = "1011"
Case "C"
    aux = "1100"
Case "D"
    aux = "1101"
Case "E"
    aux = "1110"
Case "F"
    aux = "1111"
Case Else ' Otros valores.
    Debug.Print "Dato no es hexadecimal"
End Select
trans_Binario = aux
End Function
Private Sub Form_Unload(Cancel As Integer)
Timer1.Interval = 0
CloseDatabase
mdi_total.stb_total.Panels(1).Text = "Sea Usted Bienvenido"
mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\iconos\HANDSHAK.ico")
mdi_total.stb_total.Panels(2).Text = "En Espera de otra Actividad"
End Sub
Private Sub Timer1_Timer()
If num_rec <= 20 Then
ReDim Buffer(16)
    Buffer(0) = CByte(2)
    Buffer(1) = CByte(3)
```

```

BufferSize = 2
HIDComm1.WriteTo Buffer, BufferSize
Buffer = HIDComm1.ReadFrom(BufferSize)
If BufferSize < 2 Then Exit Sub
Text5(0).Text = Buffer(0)
Text5(1).Text = Buffer(1)
If aux_orden = 2 Then
If aux_orden = 1 Then
Text3.Text = deci_bina(Trim(Text5(0).Text)) + deci_bina(Trim(Text5(1).Text))
txt_vel(1).Text = Str(bina_deci(Text3.Text))
txt_realvel.Text = Format(0.07881 * Val(txt_vel(1).Text) + 0.32, "###0.00")
aux_orden = aux_orden + 1
Exit Sub
End If
If aux_orden = 2 Then
Text4.Text = deci_bina(Trim(Text5(0).Text)) + deci_bina(Trim(Text5(1).Text))
txt_rad(0).Text = Str(bina_deci(Text4.Text))
txt_realRad.Text = Format((Val(txt_rad(0).Text) * 1000 / 1024), "###0.00")
aux_orden = aux_orden + 1
Exit Sub
End If
If aux_orden = 3 Then
Text6.Text = deci_hex(Val(Trim(Text5(0).Text)))
txt_fec(2).Text = Trim(Str(deci_hex(Val(Trim(Text5(1).Text)))) + ":" +
Trim(Str(deci_hex(Val(Trim(Text5(0).Text)))) + ":00")
aux_orden = aux_orden + 1
Timer1.Interval = 0
Exit Sub
End If
If aux_orden = 4 Then
Text6.Text = deci_hex(Val(Trim(Text5(0).Text)))
txt_fec(2).Text = Trim(Str(deci_hex(Val(Trim(Text5(1).Text)))) + "/" +
Trim(Str(deci_hex(Val(Trim(Text5(0).Text)))) + "/" + Str(Year(Now)) + " " + txt_fec(2).Text
aux_orden = 1
Timer1.Interval = 0
NUM_REG = NUM_REG + 1
num_rec = num_rec + 1
txt_barra(1).Width = 246 + txt_barra(1).Width
Text1.Text = Str(Val(Text1.Text) + 5) + "%"
Text6.Text = Str(NUM_REG)
guardar
End If
Else
MsgBox ("Transferencia Completa")
Timer1.Interval = 0
End If
my_Clear
End If
End Sub
Public Sub my_Clear()
Text3.Text = ""
Text4.Text = ""
txt_rad(0).Text = ""
txt_vel(1).Text = ""
End Sub
Private Sub OpenDatabase()

```



```

Dim dbname As String
On Error GoTo NotOpen
' Open the first workspace.
Set wks = DBEngine.Workspaces(0)
' Open the database object.
dbname = App.Path & "\monitor.mdb"
mdi_total.stb_total.Panels(1).Text = "Transfiriendo Datos"
mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\iconos\db.ico")
mdi_total.stb_total.Panels(2).Text = "Del Sistema de Monitoreo"
Set db = wks.OpenDatabase(dbname)
Exit Sub
NotOpen:
MsgBox "Error" & Str$(Err.Number) & _
    " ABRIENDO BASE DE DATOS " & dbname & "." & _
    vbCrLf & Err.Description
End Sub
' Close the database.
Private Sub CloseDatabase()
On Error GoTo NotClosed
rs_DatosR.Close
db.Close
wks.Close
Set db = Nothing
Set wks = Nothing
Exit Sub
NotClosed:
MsgBox "Error" & Str$(Err.Number) & _
    " Cerrando la Base de Datos. " & vbCrLf & _
    Err.Description
End Sub
Private Sub guardar()
If txt_rad(0).Text = "" Or txt_vel(1).Text = "" Or txt_fec(2).Text = "" Then
MsgBox "Por Favor Ingrese la Información Completa", vbInformation, "Advertencia"
Else
Dim query As String
'codigo sql
query = "INSERT INTO datos VALUES("
query = query & """" & txt_realRad.Text & ", "
query = query & """" & txt_realvel.Text & ", "
query = query & """" & txt_fec(2).Text & """"
query = query & ")"
' Ejecutando la consulta
On Error GoTo NotInserted
db.Execute query
' MsgBox "Registro Grabado Exitosamente"
Exit Sub
NotInserted:
MsgBox "Error al Grabar en BD Monitor " & Str$(Err.Number) & _
    " Añadiendo Registros" & vbCrLf & _
    Err.Description
End If
End Sub

FORM ABOUT
Option Explicit
' Opciones de seguridad de claves del Registro...

```

```

Const READ_CONTROL = &H20000
Const KEY_QUERY_VALUE = &H1
Const KEY_SET_VALUE = &H2
Const KEY_CREATE_SUB_KEY = &H4
Const KEY_ENUMERATE_SUB_KEYS = &H8
Const KEY_NOTIFY = &H10
Const KEY_CREATE_LINK = &H20
Const KEY_ALL_ACCESS = KEY_QUERY_VALUE + KEY_SET_VALUE + _
    KEY_CREATE_SUB_KEY + KEY_ENUMERATE_SUB_KEYS + _
    KEY_NOTIFY + KEY_CREATE_LINK + READ_CONTROL
' Tipos principales de claves del Registro...
Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1          ' Cadena Unicode terminada en Null
Const REG_DWORD = 4      ' Número de 32 bits
Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"
Private Declare Function RegOpenKeyEx Lib "advapi32" Alias "RegOpenKeyExA" (ByVal hKey As Long,
ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As Long, ByRef phkResult As Long)
As Long
Private Declare Function RegQueryValueEx Lib "advapi32" Alias "RegQueryValueExA" (ByVal hKey As Long,
ByVal lpValueName As String, ByVal lpReserved As Long, ByRef lpType As Long, ByVal lpData As String,
ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long) As Long
Private Sub cmdSysInfo_Click()
    Call StartSysInfo
End Sub
Private Sub cmdOK_Click()
    Unload Me
End Sub
Private Sub Form_Load()
    lblVersion.Caption = "Versión " & App.Major & "." & App.Minor & "." & App.Revision
End Sub
Public Sub StartSysInfo()
    On Error GoTo SysInfoErr
    Dim rc As Long
    Dim SysInfoPath As String
    ' Prueba a obtener del Registro la información del sistema sobre el nombre y la ruta del programa...
    If GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFO, gREGVALSYSINFO, SysInfoPath)
Then
        ' Prueba a obtener del Registro la información del sistema sobre la ruta del programa...
        ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC, gREGVALSYSINFOLOC,
SysInfoPath) Then
            ' Comprueba la existencia de una versión conocida de un archivo de 32 bits
            If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
                SysInfoPath = SysInfoPath & "\MSINFO32.EXE"
            ' Error - Imposible encontrar el archivo...
            Else
                GoTo SysInfoErr
            End If
            ' Error - Imposible encontrar la entrada de Registro...
        Else
            GoTo SysInfoErr
        End If
    End Sub

```

```

    Call Shell(SysInfoPath, vbNormalFocus)
    Exit Sub
SysInfoErr:
    MsgBox "System Information Is Unavailable At This Time", vbOKOnly
End Sub
Public Function GetKeyValue(KeyRoot As Long, KeyName As String, SubKeyRef As String, ByRef KeyVal As
String) As Boolean
    Dim i As Long                ' Contador de bucle
    Dim rc As Long               ' Código de retorno
    Dim hKey As Long            ' Controlador a una clave de Registro abierta
    Dim hDepth As Long          '
    Dim KeyValType As Long      ' Tipo de dato de una clave de Registro
    Dim tmpVal As String        ' Almacén temporal de una valor de clave de Registro
    Dim KeyValSize As Long      ' Tamaño de la variable de la clave de Registro
    '-----
    ' Abre la clave de Registro en la raíz {HKEY_LOCAL_MACHINE...}
    '-----
    rc = RegOpenKeyEx(KeyRoot, KeyName, 0, KEY_ALL_ACCESS, hKey) ' Abre la clave de Registro
    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError ' Trata el error...
    tmpVal = String$(1024, 0) ' Asigna espacio para la variable
    KeyValSize = 1024 ' Marca el tamaño de la variable
    '-----
    ' Recupera valores de claves de Registro...
    '-----
    rc = RegQueryValueEx(hKey, SubKeyRef, 0, _
        KeyValType, tmpVal, KeyValSize) ' Obtiene o crea un valor de clave
    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError ' Trata el error
    If (Asc(Mid(tmpVal, KeyValSize, 1)) = 0) Then ' Win95 agrega una cadena terminada en Null...
        tmpVal = Left(tmpVal, KeyValSize - 1) ' Se encontró Null, se extrae de la cadena
    Else ' WinNT no tiene una cadena terminada en Null...
        tmpVal = Left(tmpVal, KeyValSize) ' No se encontró Null, sólo se extrae la cadena
    End If
    '-----
    ' Determina el tipo de valor de la clave para conversión...
    '-----
    Select Case KeyValType ' Busca tipos de datos...
    Case REG_SZ ' Tipo de dato de la cadena de la clave de Registro
        KeyVal = tmpVal ' Copia el valor de la cadena
    Case REG_DWORD ' El tipo de dato de la cadena de la clave es Double Word
        For i = Len(tmpVal) To 1 Step -1 ' Convierte cada byte
            KeyVal = KeyVal + Hex(Asc(Mid(tmpVal, i, 1))) ' Genera el valor carácter a carácter
        Next
        KeyVal = Format$("&h" + KeyVal) ' Convierte Double Word a String
    End Select
    GetKeyValue = True ' Vuelve con éxito
    rc = RegCloseKey(hKey) ' Cierra la clave de Registro
    Exit Function ' Salir
GetKeyError: ' Restaurar después de que ocurra un error...
    KeyVal = "" ' Establece el valor de retorno para una cadena vacía
    GetKeyValue = False ' Devuelve un error
    rc = RegCloseKey(hKey) ' Cierra la clave de Registro
End Function
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
    picIcon(1).Visible = False
    picIcon(0).Visible = True
End Sub

```

```

Private Sub picIcon_MouseMove(Index As Integer, Button As Integer, Shift As Integer, X As Single, Y As Single)
picIcon(1).Visible = True
picIcon(0).Visible = False
End Sub

```

FORM DATOS

```

Dim numero As Integer
Private Sub Command1_Click()
frm_gestion.Show
frm_datos.Hide
End Sub
Private Sub Form_Load()
mdi_total.stb_total.Panels(1).Text = "Información de la Base"
mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\\iconos\\FILES03B.ico")
With LvwPrd
.ColumnHeaders.Add , , "Número"
.Font.Bold = True
.Font.Size = 9
.ColumnHeaders.Add , , "Radiación Solar"
.ColumnHeaders(2).Alignment = lvwColumnCenter
.ColumnHeaders(2).Width = 2000
.ColumnHeaders.Add , , "Velocidad del Viento"
.ColumnHeaders(3).Alignment = lvwColumnCenter
.ColumnHeaders(3).Width = 2500
.ColumnHeaders.Add , , "Fecha"
.ColumnHeaders(4).Width = 3000
.ColumnHeaders(4).Alignment = lvwColumnCenter
.View = lvwReport
End With
MostrarProductos
mdi_total.stb_total.Panels(2).Text = "Registros totales recibidos = " + Str(numero - 1)
End Sub
Private Sub MostrarProductos()
numero = 1
Dim RsPrd As New ADODB.Recordset
RsPrd.CursorLocation = adUseClient
RsPrd.Open "SELECT * FROM DATOS", CN, adOpenKeyset, adLockOptimistic, adCmdUnknown
LvwPrd.ListItems.Clear
While Not RsPrd.EOF
LvwPrd.ListItems.Add RsPrd.AbsolutePosition, "COD" & RsPrd!COD, RsPrd!COD
LvwPrd.ListItems.Add RsPrd.AbsolutePosition, "num" & Str(numero), Str(numero)
LvwPrd.ListItems(RsPrd.AbsolutePosition).ListSubItems.Add Key:="RADIACION", Text:=RsPrd!radiacion
LvwPrd.ListItems(RsPrd.AbsolutePosition).ListSubItems.Add Key:="VELOCIDAD",
Text:=RsPrd!velocidad
LvwPrd.ListItems(RsPrd.AbsolutePosition).ListSubItems.Add Key:="FECHA", Text:=RsPrd!Fecha
RsPrd.MoveNext
numero = numero + 1
Wend
RsPrd.Requery
RsPrd.Close
Set RsPrd = Nothing
End Sub
Private Sub Form_Unload(Cancel As Integer)
CN.Close
mdi_total.stb_total.Panels(1).Text = "Sea Usted Bienvenido"

```

```

mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\iconos\HANDSHAK.ico")
mdi_total.stb_total.Panels(2).Text = "En Espera de otra Actividad"
End Sub

```

FORM GESTION

```

Option Explicit
Private wks As Workspace
Private db As Database
Private rs_DatosR As Recordset
Dim cadena As String
Private Sub cmd_anterior_Click()
Anterior
End Sub
Private Sub cmd_borrar_Click()
Gestion_Bloqueada
Borrar
End Sub
Private Sub cmd_cancelar_Click()
Gestion_Bloqueada
End Sub
Private Sub cmd_Editar_Click()
Gestion_Bloqueada
Editar
End Sub
Private Sub cmd_graf_Click()
frm_grafrad.Show
frm_gestion.Hide
End Sub
Private Sub cmd_inicio_Click()
Carga_Datos_Gestion
End Sub
Private Sub cmd_nuevo_Click()
Blanquear_Textos
Gestion_Activa
'Call Bloqueo_Navegacion
End Sub
Private Sub cmd_rep_Click()
'frm_datos.Show
Main
frm_gestion.Hide
End Sub
Private Sub cmd_siguiete_Click()
Siguiete
End Sub
Private Sub cmd_ultimo_Click()
Ultimo
End Sub
Private Sub Form_Activate()
Set Forma_Activa = frm_gestion
Gestion_Bloqueada
Carga_Datos_Gestion
End Sub
Private Sub Form_Load()
OpenDatabase
cadena = "select * from datos"
Set rs_DatosR = db.OpenRecordset(cadena)

```

```

End Sub
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
'Actual
End Sub
Private Sub Form_Unload(Cancel As Integer)
CloseDatabase
mdi_total.stb_total.Panels(1).Text = "Sea Usted Bienvenido"
mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\\iconos\HANDSHAK.ico")
mdi_total.stb_total.Panels(2).Text = "En Espera de otra Actividad"
End Sub
' Open the database.
Private Sub OpenDatabase()
Dim dbname As String
On Error GoTo NotOpen
' Open the first workspace.
Set wks = DBEngine.Workspaces(0)
' Open the database object.
dbname = App.Path & "\\monitor.mdb"
mdi_total.stb_total.Panels(1).Text = dbname
mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\\iconos\db.ico")
mdi_total.stb_total.Panels(2).Text = "Cargando Datos a la Base de Datos"
Set db = wks.OpenDatabase(dbname)
Exit Sub
NotOpen:
MsgBox "Error" & Str$(Err.Number) & _
" ABRIENDO BASE DE DATOS " & dbname & "." & _
vbCrLf & Err.Description
End Sub
' Close the database.
Private Sub CloseDatabase()
On Error GoTo NotClosed
rs_DatosR.Close
db.Close
wks.Close
Set db = Nothing
Set wks = Nothing
Exit Sub
NotClosed:
MsgBox "Error" & Str$(Err.Number) & _
" Cerrando la Base de Datos. " & vbCrLf & _
Err.Description
End Sub
' Ingresando Registro a la Base de Datos
Private Sub cmd_guardar_Click()
If txt_rad(0).Text = "" Or txt_vel(1).Text = "" Or txt_fec(2).Text = "" Then
MsgBox "Por Favor Ingrese la Información Completa", vbInformation, "Advertencia"
Else
Dim query As String
query = "INSERT INTO datos VALUES("
query = query & """" & txt_rad(0).Text & ","
query = query & """" & txt_vel(1).Text & ","
query = query & """" & txt_fec(2).Text & """"
query = query & ")"
' Ejecutando la consulta
On Error GoTo NotInserted
db.Execute query

```

```
    MsgBox "Registro Grabado Exitosamente"
    Exit Sub
NotInserted:
    MsgBox "Error al Grabar en BD Monitor " & Str$(Err.Number) & _
        " Añadiendo Registros" & vbCrLf & _
        Err.Description
    End If
    Gestion_Bloqueada
End Sub
Private Sub txt_fec_KeyPress(Index As Integer, KeyAscii As Integer)
If KeyAscii = 13 Then
    cmd_guardar.SetFocus
End If
End Sub
Private Sub txt_rad_KeyPress(Index As Integer, KeyAscii As Integer)
If KeyAscii = 13 Then
    txt_vel(1).SetFocus
End If
End Sub
Private Sub txt_vel_KeyPress(Index As Integer, KeyAscii As Integer)
If KeyAscii = 13 Then
    txt_fec(2).SetFocus
End If
End Sub
Public Sub Carga_Datos_Gestion()
With rs_DatosR
.MoveFirst
txt_rad(0).Text = ![radiacion]
txt_vel(1).Text = ![velocidad]
txt_fec(2).Text = ![Fecha]
End With
End Sub
Public Sub Siguiente()
With rs_DatosR
.MoveNext
If Not .EOF Then
txt_rad(0).Text = ![radiacion]
txt_vel(1).Text = ![velocidad]
txt_fec(2).Text = ![Fecha]
Else
    MsgBox "Fin de Registros", vbInformation, "Advertencia"
.MoveNext
End If
End With
End Sub
Public Sub Anterior()
With rs_DatosR
.MovePrevious
If Not .BOF Then
txt_rad(0).Text = ![radiacion]
txt_vel(1).Text = ![velocidad]
txt_fec(2).Text = ![Fecha]
Else
    MsgBox "Inicio de Registros", vbInformation, "Advertencia"
.MoveNext
End If
End Sub
```

```

End With
End Sub
Public Sub Ultimo()
With rs_DatosR
.MoveLast
txt_rad(0).Text = ![radiacion]
txt_vel(1).Text = ![velocidad]
txt_fec(2).Text = ![Fecha]
End With
End Sub
Public Sub Borrar()
Dim aux As Integer
With rs_DatosR
If MsgBox("¿Esta Seguro de Borrar el Registro?", vbYesNo, "Wait Moment") = vbYes Then
.Delete
.Requery
End If
End With
End Sub
Public Sub Editar()
Dim aux As Integer
If txt_rad(0).Text = "" Or txt_vel(1).Text = "" Or txt_fec(2).Text = "" Then
MsgBox "Imposible Editar sin Reg. activo", vbInformation, "Advertencia"
Else
With rs_DatosR
.Edit
![radiacion] = Trim(txt_rad(0).Text)
![velocidad] = Trim(txt_vel(1).Text)
![Fecha] = Trim(txt_fec(2).Text)
.Update
.Requery
End With
End If
End Sub

```

FORM GRAF

```

Private Const NUM_TESTS = 5
Private Sub Grafico()
Dim test As Integer
Dim lft As Single
Dim i As Integer
Dim the_score As Integer
Dim txt As String
Dim font_hgt As Single
Dim student As Integer
' Make a convenient scale.
lft = -NUM_TESTS / 10
picScore.Scale (1 + lft, 110)-(NUM_TESTS + 0.25, -10)
font_hgt = picScore.TextHeight("X")
' Draw the axes.
picScore.Line (1, 100)-(1, 0)
picScore.Line -(NUM_TESTS, 0)
For i = 10 To 100 Step 10
picScore.Line (1 - lft / 6, i)-Step(lft / 3, 0)
txt = Format$(i)
picScore.CurrentX = 1 + lft / 4 - picScore.TextWidth(txt)

```



```

    picScore.CurrentY = i - font_hgt / 2
    picScore.Print txt
Next i
font_hgt = font_hgt * 1.2
picScore.DrawStyle = vbDot
For test = 1 To NUM_TESTS
    picScore.Line (test, 0)-(test, 100)
    txt = Format$(test)
    picScore.CurrentX = test - picScore.TextWidth(txt) / 2
    picScore.CurrentY = font_hgt / 3
    picScore.Print txt
Next test
picScore.DrawStyle = vbSolid
End Sub
Private Sub Form_Load()
    Grafico
End Sub
Private Sub Form_Unload(Cancel As Integer)
    frm_gestion.Show
End Sub

```

FORM GRAFRAD

```

Private Const MARGIN_SIZE = 60 'En Twips
Private Const SHAPE_COMMAND = "SHAPE {select RADIACION,FECHA from DATOS Order by FECHA}
AS ChildCommand COMPUTE ChildCommand, MAX(ChildCommand.[RADIACION]) AS [RADIACION]
BY [FECHA]"
Private Const CONNECT_STRING = "PROVIDER=MSDataShape;dsn=ODBC_ANDRES;uid=;pwd=;Data
Provider=MSDASQL"
Private Const FIELD_X = "FECHA"
Private Const FIELD_Y = "RADIACION"
Private Const FIELD_Z = ""
Private Const VBERR_INVALID_PROCEDURE_CALL = 5
Private Const MARKERS_VISIBLE = -1
Private Const BRACKET_LEFT = "["
Private Const BRACKET_RIGHT = "]"
Private Const SPACE_CHAR = " "
    Dim conShape As ADODB.Connection
    Dim recShape As ADODB.Recordset
Private Sub cmdClose_Click()
    Unload Me
End Sub
'-----
'Propósito: Muestra un mensaje de error al usuario
'En:
' [oError]
' Objeto Error que contiene información de error
'-----
Private Sub DisplayError(oError As ErrObject)
    MsgBox oError.Description, vbExclamation, App.Title
End Sub
Private Sub Form_Load()
    mdi_total.stb_total.Panels(1).Text = "Estadística de Radiación Solar"
    mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\iconos\ifa.ico")
    mdi_total.stb_total.Panels(2).Text = "Formato disperso"
    On Error GoTo Form_Load_Error
    'Crear una conexión abierta al proveedor de Data Shape

```

```

Set conShape = New ADODB.Connection
conShape.Open CONNECT_STRING
'Crear y abrir un recordset
Set recShape = New ADODB.Recordset
recShape.Open SHAPE_COMMAND, conShape
'Rellenar el gráfico con los datos de recordset
ShowRecordsInChart FIELD_X, FIELD_Y, FIELD_Z
'Mostrar u ocultar marcadores
ShowMarkers MARKERS_VISIBLE
Exit Sub
Form_Load_Error:
  DisplayError Err
  Exit Sub
End Sub
Private Sub Form_Resize()
  Dim sngButtonTop As Single
  Dim sngScaleWidth As Single
  Dim sngScaleHeight As Single
  On Error GoTo Form_Resize_Error
  With Me
    sngScaleWidth = .ScaleWidth
    sngScaleHeight = .ScaleHeight
    'Mover el botón Cerrar a la esquina inferior derecha
    With .cmdClose
      sngButtonTop = sngScaleHeight - (.Height + MARGIN_SIZE)
      .Move sngScaleWidth - (.Width + MARGIN_SIZE), sngButtonTop
    End With
    .chtReport.Move MARGIN_SIZE, _
      MARGIN_SIZE, _
      sngScaleWidth - (2 * MARGIN_SIZE), _
      sngButtonTop - (2 * MARGIN_SIZE)
  End With
  Exit Sub
Form_Resize_Error:
  'Se producirá un error si el usuario reduce tanto el tamaño
  'del formulario que se calculan altos y anchos
  'negativos
  Resume Next
End Sub
'-----
'Propósito:  determina si la clave pasada se usa en la colección
'           pasada.
'En:
'[cCol]  La colección en la que se va a comprobar si se usa la clave.
'[sKey]  La clave que se busca.
'Devuelve:  Si la colección usa la clave, devuelve verdadero.
'           En caso contrario, devuelve falso.
'-----
Private Function IsKeyInCollection(cCol As Collection, sKey As String) As Boolean
  Dim v As Variant
  On Error Resume Next
  v = cCol.Item(sKey)
  'Es importante comprobar el error 5 en lugar de comprobar
  'cualquier error, porque un error puede ocurrir incluso si la clave es válida.
  'Si la clave existía pero estaba asociada a un elemento que
  'era un objeto, se produciría un error porque no se usó 'Set'

```

```

'para asignarla a 'v'.
IsKeyInCollection = (Err.Number <> VBERR_INVALID_PROCEDURE_CALL)
Err.Clear
End Function
'-----
'Propósito: Muestra u oculta marcadores de serie, según
'           el parámetro.
'En:
'[bShow] Si es True, se mostrarán todos los marcadores de serie.
'         En caso contrario, se ocultarán todos los marcadores de serie.
'-----
Private Sub ShowMarkers(bShow As Boolean)
    Dim i As Long
    On Error GoTo ShowMarkers_Click_Error
    With chtReport.Plot
        For i = 1 To .SeriesCollection.Count
            .SeriesCollection(i).SeriesMarker.Show = bShow
        Next
    End With
    Exit Sub
ShowMarkers_Click_Error:
    DisplayError Err
    Exit Sub
End Sub
'-----
'Propósito: Muestra los datos resumidos del recordset pasado
'           en el gráfico.
'En:
'[recShape]
'           Un recordset creado con un comando Shape que agrupa
'           por uno o dos campos y resume uno.
'[sFldX]
'           El nombre del campo por el que se agrupa en el eje X.
'[sFldY]
'           El nombre del campo que se resume en el eje Y.
'[sFldZ]
'           El nombre del campo por el que se agrupa en el eje Z. Este
'           campo debe ser una cadena de longitud cero, si el recordset
'           sólo agrupa por un campo.
'-----
Private Sub ShowRecordsInChart(sFldX As String, _
                             sFldY As String, _
                             sFldZ As String)
    Dim bUseZ As Boolean
    Dim cRows As Collection
    Dim cCols As Collection
    Dim lCol As Long
    Dim lRow As Long
    Dim lMaxCol As Long
    Dim lMaxRow As Long
    Dim sValue As String
    On Error GoTo ShowRecordsInChart_Error
    If Len(sFldZ) = 0 Then bUseZ = False Else bUseZ = True
    Set cRows = New Collection
    Set cCols = New Collection
    With Me.chtReport

```

```
'Desactivar el trazado del gráfico
.Repaint = False
With .DataGrid
  'Borrar el gráfico
  .DeleteRows 1, .RowCount
  .DeleteColumns 1, .ColumnCount
  .DeleteColumnLabels 1, .ColumnLabelCount
  .DeleteRowLabels 1, .RowLabelCount
  'Make sure there is one level of labels
  .InsertColumnLabels 1, 1
  .InsertRowLabels 1, 1
  'Si no se usa el eje Z, asegurarse de que
  'hay una columna
  If Not bUseZ Then .InsertColumns 1, 1
  recShape.MoveFirst
  Do Until recShape.EOF
    'Asegurarse de que se agrega una fila para este campo X
    sValue = FixNull(recShape.Fields(sFldX).Value, False)
    If Not IsKeyInCollection(cRows, sValue) Then
      IMaxRow = IMaxRow + 1
      IRow = IMaxRow
      'Almacenar el índice de fila asociado al
      'nombre de la fila
      cRows.Add IRow, sValue
      .InsertRows IRow, 1
      .RowLabel(IRow, 1) = sValue
    Else
      IRow = cRows.Item(sValue)
    End If
    'Asegurarse de que se agrega una columna para este campo Z
    If bUseZ Then
      sValue = FixNull(recShape.Fields(sFldZ).Value, False)
      If Not IsKeyInCollection(cCols, sValue) Then
        IMaxCol = IMaxCol + 1
        ICol = IMaxCol
        'Almacenar el índice de columna asociado al
        'nombre de columna
        cCols.Add ICol, sValue
        .InsertColumns ICol, 1
        .ColumnLabel(ICol, 1) = sValue
      Else
        ICol = cCols.Item(sValue)
      End If
      'Establecer el valor del punto de datos para la fila de este registro y la columna
      .SetData IRow, ICol, FixNull(recShape.Fields.Item(sFldY).Value, True), 0
    Else
      'Establecer el valor del punto de datos para la fila de este registro
      'Sólo hay una columna en este caso
      .SetData IRow, 1, FixNull(recShape.Fields.Item(sFldY).Value, True), 0
    End If
    'Mover el recordset al siguiente registro
    recShape.MoveNext
  Loop
End With
'Activar el trazado del gráfico
.Repaint = True
```

```

    End With
    Exit Sub
ShowRecordsInChart_Error:
    'Asegurarse de que se vuelve a activar el trazado de gráficos
    Me.chtReport.Repaint = True
    DisplayError Err
    Exit Sub
End Sub
'-----
'Propósito: Comprueba si un valor Variant es nulo. Si el valor es nulo,
'           devuelve vbNullString o cero.
'En:
'[vField]
'           El valor Variant que se va a comprobar.
'[bNumericRequired]
'           Si es True, devuelve 0 si el valor Variant es nulo. En caso contrario,
'           devuelve vbNullString.
'-----
Private Function FixNull(vField As Variant, _
                        bNumericRequired As Boolean) As Variant
    If IsNull(vField) Then
        If bNumericRequired Then
            FixNull = 0
        Else
            FixNull = vbNullString
        End If
    Else
        FixNull = vField
    End If
End Function
Private Sub Form_Unload(Cancel As Integer)
    mdi_total.stb_total.Panels(1).Text = "Sea Usted Bienvenido"
    mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\iconos\HANDSHAK.ico")
    mdi_total.stb_total.Panels(2).Text = "En Espera de otra Actividad"
End Sub

```

FORM GRAFVEL

```

Private Const MARGIN_SIZE = 60 'En Twips
Private Const SHAPE_COMMAND = "SHAPE {select VELOCIDAD,FECHA from DATOS Order by
VELOCIDAD} AS ChildCommand COMPUTE ChildCommand, MAX(ChildCommand.[VELOCIDAD]) AS
[VELOCIDAD] BY [FECHA]"
Private Const CONNECT_STRING = "PROVIDER=MSDataShape;dsn=ODBC_ANDRES;uid=;pwd=;Data
Provider=MSDASQL"
Private Const FIELD_X = "FECHA"
Private Const FIELD_Y = "VELOCIDAD"
Private Const FIELD_Z = ""
Private Const VBERR_INVALID_PROCEDURE_CALL = 5
Private Const MARKERS_VISIBLE = -1
Private Const BRACKET_LEFT = "["
Private Const BRACKET_RIGHT = "]"
Private Const SPACE_CHAR = " "
Dim conShape As ADODB.Connection
Dim recShape As ADODB.Recordset
Private Sub cmdClose_Click()
    Unload Me
End Sub

```

```

'-----
'Propósito: Muestra un mensaje de error al usuario
'En:
' [oError]
' Objeto Error que contiene información de error
'-----
Private Sub DisplayError(oError As ErrObject)
    MsgBox oError.Description, vbExclamation, App.Title
End Sub
Private Sub Form_Load()
    mdi_total.stb_total.Panels(1).Text = "Estadística Velocidad de Viento"
    mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\iconos\ct.ico")
    mdi_total.stb_total.Panels(2).Text = "En Formato Disperso"
    On Error GoTo Form_Load_Error
    'Crear una conexión abierta al proveedor de Data Shape
    Set conShape = New ADODB.Connection
    conShape.Open CONNECT_STRING
    'Crear y abrir un recordset
    Set recShape = New ADODB.Recordset
    recShape.Open SHAPE_COMMAND, conShape
    'Rellenar el gráfico con los datos de recordset
    ShowRecordsInChart FIELD_X, FIELD_Y, FIELD_Z
    'Mostrar u ocultar marcadores
    ShowMarkers MARKERS_VISIBLE
Exit Sub
Form_Load_Error:
    DisplayError Err
    Exit Sub
End Sub
Private Sub Form_Resize()
    Dim sngButtonTop As Single
    Dim sngScaleWidth As Single
    Dim sngScaleHeight As Single
    On Error GoTo Form_Resize_Error
    With Me
        sngScaleWidth = .ScaleWidth
        sngScaleHeight = .ScaleHeight
        'Mover el botón Cerrar a la esquina inferior derecha
        With .cmdClose
            sngButtonTop = sngScaleHeight - (.Height + MARGIN_SIZE)
            .Move sngScaleWidth - (.Width + MARGIN_SIZE), sngButtonTop
        End With
        .chtReport.Move MARGIN_SIZE, _
            MARGIN_SIZE, _
            sngScaleWidth - (2 * MARGIN_SIZE), _
            sngButtonTop - (2 * MARGIN_SIZE)
    End With
Exit Sub
Form_Resize_Error:
    'Se producirá un error si el usuario reduce tanto el tamaño
    'del formulario que se calculan altos y anchos
    'negativos
    Resume Next
End Sub
'-----
'Propósito: determina si la clave pasada se usa en la colección

```

```

'      pasada.
'En:
'[cCol]  La colección en la que se va a comprobar si se usa la clave.
'[sKey]  La clave que se busca.
'Devuelve: Si la colección usa la clave, devuelve verdadero.
'      En caso contrario, devuelve falso.
'-----
Private Function IsKeyInCollection(cCol As Collection, sKey As String) As Boolean
    Dim v As Variant
    On Error Resume Next
    v = cCol.Item(sKey)
    'Es importante comprobar el error 5 en lugar de comprobar
    'cualquier error, porque un error puede ocurrir incluso si la clave es válida.
    'Si la clave existía pero estaba asociada a un elemento que
    'era un objeto, se produciría un error porque no se usó 'Set'
    'para asignarla a 'v'.
    IsKeyInCollection = (Err.Number <> VBERR_INVALID_PROCEDURE_CALL)
    Err.Clear
End Function
'-----
'Propósito: Muestra u oculta marcadores de serie, según
'      el parámetro.
'En:
'[bShow] Si es True, se mostrarán todos los marcadores de serie.
'      En caso contrario, se ocultarán todos los marcadores de serie.
'-----
Private Sub ShowMarkers(bShow As Boolean)
    Dim i As Long
    On Error GoTo ShowMarkers_Click_Error
    With chtReport.Plot
        For i = 1 To .SeriesCollection.Count
            .SeriesCollection(i).SeriesMarker.Show = bShow
        Next
    End With
    Exit Sub
ShowMarkers_Click_Error:
    DisplayError Err
    Exit Sub
End Sub
'-----
'Propósito: Muestra los datos resumidos del recordset pasado
'      en el gráfico.
'En:
'[recShape]
'      Un recordset creado con un comando Shape que agrupa
'      por uno o dos campos y resume uno.
'[sFldX]
'      El nombre del campo por el que se agrupa en el eje X.
'[sFldY]
'      El nombre del campo que se resume en el eje Y.
'[sFldZ]
'      El nombre del campo por el que se agrupa en el eje Z. Este
'      campo debe ser una cadena de longitud cero, si el recordset
'      sólo agrupa por un campo.
'-----
Private Sub ShowRecordsInChart(sFldX As String, _

```

```

                sFldY As String, _
                sFldZ As String)
Dim bUseZ As Boolean
Dim cRows As Collection
Dim cCols As Collection
Dim lCol As Long
Dim lRow As Long
Dim lMaxCol As Long
Dim lMaxRow As Long
Dim sValue As String
On Error GoTo ShowRecordsInChart_Error
If Len(sFldZ) = 0 Then bUseZ = False Else bUseZ = True
Set cRows = New Collection
Set cCols = New Collection
With Me.chtReport
    'Desactivar el trazado del gráfico
    .Repaint = False
    With .DataGrid
        'Borrar el gráfico
        .DeleteRows 1, .RowCount
        .DeleteColumns 1, .ColumnCount
        .DeleteColumnLabels 1, .ColumnLabelCount
        .DeleteRowLabels 1, .RowLabelCount
        'Make sure there is one level of labels
        .InsertColumnLabels 1, 1
        .InsertRowLabels 1, 1
        'Si no se usa el eje Z, asegurarse de que
        'hay una columna
        If Not bUseZ Then .InsertColumns 1, 1
        recShape.MoveFirst
        Do Until recShape.EOF
            'Asegurarse de que se agrega una fila para este campo X
            sValue = FixNull(recShape.Fields(sFldX).Value, False)
            If Not IsKeyInCollection(cRows, sValue) Then
                lMaxRow = lMaxRow + 1
                lRow = lMaxRow
                'Almacenar el índice de fila asociado al
                'nombre de la fila
                cRows.Add lRow, sValue
                .InsertRows lRow, 1
                .RowLabel(lRow, 1) = sValue
            Else
                lRow = cRows.Item(sValue)
            End If
            'Asegurarse de que se agrega una columna para este campo Z
            If bUseZ Then
                sValue = FixNull(recShape.Fields(sFldZ).Value, False)
                If Not IsKeyInCollection(cCols, sValue) Then
                    lMaxCol = lMaxCol + 1
                    lCol = lMaxCol
                    'Almacenar el índice de columna asociado al
                    'nombre de columna
                    cCols.Add lCol, sValue
                    .InsertColumns lCol, 1
                    .ColumnLabel(lCol, 1) = sValue
                Else

```



```

        ICol = cCols.Item(sValue)
    End If
    'Establecer el valor del punto de datos para la fila de este registro y la columna
    .SetData IRow, ICol, FixNull(recShape.Fields.Item(sFldY).Value, True), 0
Else
    'Establecer el valor del punto de datos para la fila de este registro
    'Sólo hay una columna en este caso
    .SetData IRow, 1, FixNull(recShape.Fields.Item(sFldY).Value, True), 0
End If
'Mover el recordset al siguiente registro
recShape.MoveNext
Loop
End With
'Activar el trazado del gráfico
.Repaint = True
End With
Exit Sub
ShowRecordsInChart_Error:
'Asegurarse de que se vuelve a activar el trazado de gráficos
Me.chtReport.Repaint = True
DisplayError Err
Exit Sub
End Sub
'-----
'Propósito: Comprueba si un valor Variant es nulo. Si el valor es nulo,
' devuelve vbNullString o cero.
'En:
'[vField]
' El valor Variant que se va a comprobar.
'[bNumericRequired]
' Si es True, devuelve 0 si el valor Variant es nulo. En caso contrario,
' devuelve vbNullString.
'-----
Private Function FixNull(vField As Variant, _
    bNumericRequired As Boolean) As Variant
    If IsNull(vField) Then
        If bNumericRequired Then
            FixNull = 0
        Else
            FixNull = vbNullString
        End If
    Else
        FixNull = vField
    End If
End Function
Private Sub Form_Unload(Cancel As Integer)
mdi_total.stb_total.Panels(1).Text = "Sea Usted Bienvenido"
mdi_total.stb_total.Panels(1).Picture = LoadPicture(App.Path & "\iconos\HANDSHAK.ico")
mdi_total.stb_total.Panels(2).Text = "En Espera de otra Actividad"
End Sub

```

FORM INICIO

```

Option Explicit
Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
    mdi_total.Show

```

```

End Sub
Private Sub Form_Load()
    Timer1.Interval = 300
End Sub
Private Sub Frame1_Click()
    Timer1.Interval = 0
    Unload Me
End Sub
Private Sub Timer1_Timer()
If imgLogo(3).Visible = True Then
    imgLogo(3).Visible = False
    imgLogo(0).Visible = True
ElseIf imgLogo(0).Visible = True Then
    imgLogo(0).Visible = False
    imgLogo(1).Visible = True
ElseIf imgLogo(1).Visible = True Then
    imgLogo(1).Visible = False
    imgLogo(2).Visible = True
ElseIf imgLogo(2).Visible = True Then
    imgLogo(2).Visible = False
    imgLogo(3).Visible = True
End If
End Sub

```

FORM MDI TOTAL

```

Private Sub MDIForm_Load()
Fecha = Date
Hora = Time
stb_total.Panels(1).Width = 4500
stb_total.Panels(1).Alignment = sbrCenter
stb_total.Font.Bold = True
stb_total.Panels(1).Text = "Sea Usted Bienvenido"
stb_total.Font.Size = 10
stb_total.Panels(2).Width = 6000
stb_total.Panels(2).Alignment = sbrCenter
stb_total.Panels(2).Text = "Conecte al puerto USB par recibir Datos"
stb_total.Panels(3).Width = 1710
stb_total.Panels(4).Width = 3000
stb_total.Panels(4).Alignment = sbrCenter
Actual
End Sub
Private Sub MDIForm_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Actual
End Sub
Private Sub mnu_acerca_Click()
frm_About.Show
Actual
End Sub
Private Sub mnu_cargar_Click()
frm_gestion.Show
Actual
End Sub
Private Sub mnu_cascade_Click()
mdi_total.Arrange vbCascade
End Sub
Private Sub mnu_grafad_Click()

```

```

frm_grafrad.Show
End Sub
Private Sub mnu_grafvel_Click()
frm_grafvel.Show
Actual
End Sub
Private Sub mnu_horizontal_Click()
mdi_total.Arrange vbTileHorizontal
End Sub
Private Sub mnu_organizar_Click()
mdi_total.Arrange vbArrangeIcons
End Sub
Private Sub mnu_salir_Click()
End
End Sub
Private Sub mnu_trans_Click()
Form1.Show
End Sub
Private Sub mnu_ventana_Click()
'mdi_total.Show modal, cascade
End Sub
Private Sub mnu_ver_Click()
Main
Actual
End Sub
Private Sub Actual()
stb_total.Panels(4).Text = FechayHora
End Sub
Private Sub mnu_vertical_Click()
Me.Arrange vbTileVertical
End Sub

```

MODULOS

GLOBAL

```

Option Explicit
Public CN As New ADODB.Connection
Public Forma_Activa As Form
Dim Fecha As String
Dim Hora As String
Public Sub Main()
Set CN = New ADODB.Connection
CN.Open "DSN=odbc_andres"
frm_datos.Show
End Sub
Public Sub Activo_Navegacion()
frm_gestion.cmd_inicio.Enabled = True
frm_gestion.cmd_anterior.Enabled = True
frm_gestion.cmd_siguiente.Enabled = True
frm_gestion.cmd_ultimo.Enabled = True
End Sub
Public Sub Bloqueo_Navegacion()
frm_gestion.cmd_inicio.Enabled = False
frm_gestion.cmd_anterior.Enabled = False
frm_gestion.cmd_siguiente.Enabled = False

```

```

frm_gestion.cmd_ultimo.Enabled = False
End Sub
Public Sub Blanquear_Textos()
frm_gestion.txt_fec(2).Text = ""
frm_gestion.txt_rad(0).Text = ""
frm_gestion.txt_vel(1).Text = ""
frm_gestion.txt_rad(0).SetFocus
End Sub
'Activo nuevo,modificar,eliminar
'Desactivo grabar,cancelar
Public Sub Gestion_Activa()
frm_gestion.cmd_nuevo.Enabled = False
frm_gestion.cmd_Editar.Enabled = False
frm_gestion.cmd_borrar.Enabled = False
frm_gestion.cmd_guardar.Enabled = True
frm_gestion.cmd_cancelar.Enabled = True
End Sub
'Activo grabar,cancelar
'Desactivo nuevo,modificar,eliminar
Public Sub Gestion_Bloqueada()
frm_gestion.cmd_nuevo.Enabled = True
frm_gestion.cmd_Editar.Enabled = True
frm_gestion.cmd_borrar.Enabled = True
frm_gestion.cmd_guardar.Enabled = False
frm_gestion.cmd_cancelar.Enabled = False
End Sub
Public Sub Carga_Datos_Gestion()
End Sub
Public Function FechayHora() As String
Fecha = Date
Hora = Time
FechayHora = Fecha + " " + Hora
End Function
Public Function deci_bina(n As Integer) As String
Dim C As Double
Dim J As Integer, i As Integer
Dim D(8) As Integer
Dim cadena As String
Dim l As Integer
C = n
J = 1
While C >= 2
D(J) = RESIDUO(C)
C = C \ 2
J = J + 1
Wend
D(J) = C
cadena = ""
For i = J To 1 Step -1
cadena = cadena + Trim(Str(D(i)))
Next i
cadena = Trim(cadena)
l = Len(cadena)
'MsgBox (Str(l))
If l < 8 Then
For i = 1 To 8 - l Step 1

```

```

cadena = "0" + cadena
Next i
End If
deci_bina = cadena
End Function
Private Function RESIDUO(C As Double) As Integer
RESIDUO = C Mod 2
End Function
Public Function bina_deci(n As String) As Integer
Dim C As Integer, i As Integer, J As Integer, D As Integer
Dim B(1000) As Integer, E(100) As Integer
Dim num As Integer
Dim aux2 As Integer
C = Len(n)
J = 0
For i = 1 To C
B(i) = Val(Mid(n, i, 1))
E(i) = J
J = J + 1
Next i
D = J
num = 0
For i = 1 To C
If B(i) = 0 Or B(i) = 1 Then
num = num + SUMA(B(i), E(D))
D = D - 1
bina_deci = num
Else
MsgBox "DEBE SER UN NUMERO BINARIO"
'Label1 = ""
'Text1 = ""
'Text1.SetFocus
i = C
End If
Next i
End Function
Private Function SUMA(X As Integer, Y As Integer) As Integer
SUMA = (X * (2 ^ Y))
End Function
Public Function deci_hex(n As Integer) As Integer
Dim C As Double
Dim J As Integer, i As Integer
Dim D(8) As Integer
Dim cadena As String
Dim l As Integer
C = n
J = 1
While C >= 16
D(J) = RESIDUO_HEX(C)
C = C \ 16
J = J + 1
Wend
D(J) = C
cadena = ""
For i = J To 1 Step -1
cadena = cadena + Trim(Str(D(i)))

```

```
Next i
deci_hex = Val(cadena)
End Function
Private Function RESIDUO_HEX(C As Double) As Integer
RESIDUO_HEX = C Mod 16
End Function
```

ANEXO M
ANÁLISIS DE COSTOS

ANALISIS DE COSTOS

ITEM	CANT	ELEMENTOS	COSTO UNITARIO (\$)	COSTO TOTAL (\$)
SENSORES				
1	1	Sensor de Radiación Solar (Piranómetro)	100	100
2	1	Sensor de Velocidad de Viento (Anemómetro)	450	450
TOTAL				550
TARJETA ADQUISICION DE DATOS				
3	1	Placas PCB	40	40
4	1	Microcontrolador PIC 16C745/JW	10	10
5	1	Microcontrolador PIC 16F877	8	8
6	1	RTC (Reloj de tiempo real)	6	6
7	2	24LC256 (Memorias I2C)	5	10
8	1	LM324 (Amplificador Operacional)	2	2
9	1	MAX232 (Comunicación Serial)	5	5
10	1	Cristal OSC 20 MHz	0,4	0,4
11	1	Cristal OSC 6 MHz	1	1
12	1	Cristal OSC 32 MHz	1	1
13	1	Bateria 3V + sócalo	3	3
14	1	Regulador de Voltaje 5V LM7805	0,6	0,6
15	1	Transistor 2N3904	0,3	0,3
16	1	Potenciómetro 5 Kohm	1	1
17	2	Pulsador	0,5	1
18	1	Resistencia 47 Kohm	0,1	0,1
19	4	Resistencia 1 Kohm	0,1	0,4
20	3	Resistencia 10 Kohm	0,1	0,3
21	2	Resistencia 470 ohm	0,1	0,2
22	2	Resistencia 4,7 Kohm	0,1	0,2
23	1	Resistencia 330 ohm	0,1	0,1
24	2	Capacitor 100 pF	0,15	0,3
25	2	Capacitor 33 pF	0,15	0,2
26	1	Capacitor 200 uF	0,15	0,1
27	1	Capacitor 0,1 uF	0,15	0,1
28	4	Borneras	1	4
29	3	Led 5mm.	0,1	0,3
TOTAL				95,6
CASE EXTERNO				
30	1	Batería de 6V	6	6
31	1	Case	10	10
CONEXIÓN				
32	1	Cable USB	2	2
33	1	Cable Serial	2	2
TOTAL	49			665,6

ÍNDICE DE FIGURAS

Figura. 1.1. Generadores de Energía Eólica	12
Figura. 1.2. Componentes en una instalación fotovoltaica autónoma	19
Figura. 1.3. Componentes en una instalación fotovoltaica conectada a la red	20
Figura. 2.1. La tierra del ecuador se calienta más por la acción solar que la de los polos, debido a que recibe más cantidad de radiación por unidad de superficie	27
Figura. 2.2. Circuito de acondicionamiento del sensor de radiación solar	30
Figura. 2.3. Distribución de pines memoria I2C	38
Figura. 2.4. Diagrama de Conexiones del RTC	42
Figura. 2.5. MicroCode Studio Plus	45
Figura. 2.6. Ventana de comunicación serial	45
Figura 2.7. Diagrama de flujo del Código del Programa	48
Figura. 2.8. Diagrama de Bloques del Sistema de Sensores	50
Figura. 3.1. Estructura de Capas del Bus USB	54
Figura. 3.2. Esquema de un Concentrador o HUB de USB	55
Figura. 3.3. (a) Posible esquema de Conexiones del USB	56
Figura. 3.3. (b) Dispositivos USB conectados a un PC	56
Figura. 3.4. Capas del Sistema de Comunicaciones USB	57
Figura. 3.5. Esquema del cable USB	58
Figura. 3.6. Conectores USB	59
Figura. 3.7. Diagrama de flujo del Sistema de Adquisición USB	62
Figura. 3.8. Diagrama de Bloques del Sistema	64
Figura. 3.9. Ventana de selección del ActiveX para comunicación USB	66
Figura 3.10. Selección de Dispositivo USB	66
Figura. 3.11. Diagrama de flujo de los procesos de la aplicación HMI	67
Figura. 3.12. Carátula del Interfaz (HMI)	73
Figura. 3.13. Ventana de selección de actividades	73
Figura. 3.14. Barra de información	74
Figura. 3.15. Transferir Datos	74
Figura. 3.16. Gestión de Datos	75
Figura. 3.17. Ventanas de Gráficos y Base de Datos	75
Figura. 3.18. Gráfico Radiación Vs. Tiempo	77
Figura. 3.19. Gráfico Velocidad del Viento Vs. Tiempo	78

ÍNDICE DE TABLAS

Tabla. 1.1. Ventajas en un campo de 10 MW	4
Tabla. 2.1. Distribución de pines memoria I ² C	38
Tabla. 2.2. Pines de dirección memoria I ² C	39
Tabla. 2.3. Mapa de memoria del RTC	40
Tabla. 2.4. Registro de control del RTC	41
Tabla. 2.5. Frecuencia de la señal de reloj del RTC	42
Tabla. 2.6. Bytes de almacenamiento de canal, fecha y hora de la medición	43

GLOSARIO

ActiveX	Es uno de los mecanismos que usan los programas de computadoras para comunicarse con las bases de datos, darles órdenes y obtener resultados de ellas.
Aerogenerador	Es un generador de electricidad activado por la acción del viento.
Anemómetro	Es un aparato meteorológico que se usa para la predicción del tiempo y, específicamente, para medir la magnitud de la velocidad del viento.
AWG	Es la medida de diámetro de los conductores eléctricos (cables o alambres) indicados con la referencia AWG (American Wire Gauge).
Base de datos	Es un conjunto de datos que pertenecen al mismo contexto almacenados sistemáticamente para su posterior uso.
Bit	Unidad más pequeña de información, representa la unidad base en la comunicación digital mediante dos estados “0” o “1”.
Bus	Un bus es un subsistema que transfiere datos o electricidad entre componentes del ordenador dentro de un ordenador o entre ordenadores. A diferencia de una conexión punto a punto, un bus puede conectar mediante lógica varios periféricos utilizando el mismo conjunto de cables.
Controlador	Un controlador de dispositivo (llamado normalmente controlador, o, en inglés, <i>driver</i>) es un programa informático que permite al sistema operativo interactuar con un periférico, haciendo una abstracción del

hardware y proporcionando un interfaz -posiblemente estandarizada- para usarlo.

- Dínamo** Es una máquina destinada a la transformación de energía mecánica en eléctrica, produciendo corriente continua mediante el fenómeno de la inducción electromagnética producida por un imán o un electroimán rotario sobre una bobina fija o viceversa.
- FDDI** En computación, la FDDI (Fiber distributed data interface) se define como una topología de red local en doble anillo y con soporte físico de fibra óptica.
- Hardware** Comprende toda la parte física de un sistema
- Hub** En informática un *hub* o concentrador es un equipo de redes que permite conectar entre sí otros equipos y retransmite los paquetes que recibe desde cualquiera de ellos a todos los demás.
- I²C** Es un bus de comunicaciones serie. Su nombre viene de *Inter-Integrated Circuit* (Circuitos Inter-Integrados).
- Interfaz** Conexión que permite la correcta comunicación entre dos o más dispositivos
- Interrupción** También conocida como interrupción hardware, es una señal recibida por el procesador de un ordenador, indicando que debe "interrumpir" el curso de ejecución actual y pasar a ejecutar código específico para tratar esta situación.
- Irradiancia** Es la magnitud utilizada para describir la potencia incidente por unidad de superficie de todo tipo de radiación electromagnética. En unidades del sistema internacional se mide en W/m².

Mbps	Un megabit por segundo (Mbps o también Mbit/s) es una unidad que se usa para cuantificar la velocidad de transmisión de información equivalente a 1024 kilobits por segundo o 1048576 bits por segundo.
Microcontrolador	Circuito integrado o chip que incluye en su interior las tres unidades funcionales de un ordenador: CPU, Memoria y Unidades de E/S, es decir, se trata de un computador completo en un solo circuito integrado. Aunque sus prestaciones son limitadas, además de dicha integración, su característica principal es su alto nivel de especialización.
Panel solar	Es un modulo que aprovecha la energía de la radiación solar. El término comprende a los colectores solares utilizados para producir agua caliente (usualmente doméstica) y a los paneles fotovoltaicos utilizados para generar electricidad.
Parque eólico	Es una agrupación de aerogeneradores que se utilizan generalmente para la producción de energía eléctrica. Los parques eólicos se pueden situar en tierra o en el mar. El número de aerogeneradores que componen un parque es muy variable, y depende fundamentalmente de la superficie disponible y de las características del viento en el emplazamiento.
PCB	Un circuito impreso o PCB (del inglés <i>Printed Circuit Board</i>), es un medio para sostener mecánicamente y conectar eléctricamente componentes electrónicos, a través de <i>rut</i> as o <i>pistas</i> de material conductor, grabados desde hojas de cobre laminadas sobre un sustrato no conductor.
Periférico	Dispositivo externo que permite realizar funciones adicionales a un dispositivo
Piranómetro	Instrumento meteorológico utilizado para medir la radiación solar incidente en superficie. Generalmente se utilizan tres medidas de radiación: semiesférica total, difusa y directa. El principio físico

utilizado generalmente en la medida es un termopar sobre el que incide la radiación a través de dos cúpulas semiesféricas de vidrio. Las medidas se expresan en KW/m^2 .

Plug & Play	Conocida también por su abreviatura PnP, es la tecnología que permite a un dispositivo informático ser conectado a un ordenador sin tener que configurar jumpers ni proporcionar parámetros a sus controladores. Para que eso sea posible, el sistema operativo con el que funciona el ordenador debe tener soporte para dicho dispositivo. La frase <i>plug-and-play</i> se traduce como <i>enchufar y listo</i> .
Protocolo	Conjunto de reglas estandarizadas que controlan el funcionamiento y transmisión de información en un tipo de comunicación.
Radiación electromagnética	Es una combinación de campos eléctricos y magnéticos oscilantes y perpendiculares entre sí, que se propagan a través del espacio transportando energía de un lugar a otro.
Radiación solar	Es el conjunto de radiaciones electromagnéticas que son emitidas por el Sol. Éstas van desde el infrarrojo hasta el ultravioleta. La unidad práctica que describe la radiación solar que llega a la tierra es la irradiancia, o unidad de potencia por metro cuadrado [w/m^2].
Red	Grupo de dispositivos interconectados que pueden transferir información entre ellos.
Sensor	Instrumento que produce una señal, usualmente eléctrica, que refleja el valor de una propiedad, mediante alguna correlación definida (su <i>ganancia</i>), es decir que no altera la propiedad sensada.
Software	Comprende toda la parte lógica de un sistema relacionada con programas o conjunto de instrucciones grabadas en una memoria.
SPI	El bus SPI (del inglés Serial Peripheral Interface) es un estándar de comunicaciones.

Transductor	Instrumento que convierte una forma de energía en otra (o una propiedad en otra).
Turbina	Es una máquina de fluido que permite transformar la energía del fluido que la atraviesa en movimiento rotativo de un eje. El fluido puede ser líquido (o comportarse como tal) como en las turbinas hidráulicas de las centrales hidroeléctricas, o gaseoso como en las turbinas de los aerogeneradores, las turbinas de vapor y las turbinas de gas.
USART	Universal Synchronous/Asynchronous Receiver/Transmitter (transmisor/receptor síncrono/asíncrono universal)

Sangolquí, _____

ELABORADO POR:

Juan Vladimir Serrano Cevallos

Andrés Roberto Vallejo Luna

El Decano

El Secretario Académico

Ing. Xavier F. Martinez C.
TCRN. DE E.M.

Dr. Jorge Carvajal R.