SISTEMA DE MEDICION DE VARIABLES ELECTRICAS DE UN GENERADOR SINCRONO USANDO TELEMETRIA

DELBER ALBERTO ARZUZA TORRES
JOSÉ IGNACIO VILLARREAL MARIMÓN

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA
CARTAGENA DE INDIAS
2005

SISTEMA DE MEDICION DE VARIABLES ELECTRICAS DE UN GENERADOR SINCRONO USANDO TELEMETRIA

DELBER ALBERTO ARZUZA TORRES
JOSÉ IGNACIO VILLARREAL MARIMÓN

Trabajo de Grado, presentado para optar al título de Ingeniero Electrónico

Director
EDUARDO GÓMEZ VÁSQUEZ
Magíster en Ciencias Computacionales

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERIA ELECTRICA Y ELECTRONICA
CARTAGENA
2005

Cartagena de Indias D.T y C. Septiembre de 2005

Señores:
Comité Evaluador
Departamento de Ingeniería Eléctrica y Electrónica
Apreciados señores:

Por medio de la presente me permito informarles que el trabajo de grado titulado "**SISTEMA DE MEDICION DE VARIABLES ELECTRICAS DE UN GENERADOR SINCRONO USANDO TELEMETRIA"** ha sido desarrollado de acuerdo a los objetivos establecidos.

Como autores del proyecto consideramos que el trabajo es satisfactorio y amerita ser presentado para su evaluación.

Atentamente,

_____	_____
José I. Villarreal Marimón.	Delber A. Arzuza Torres

Cartagena de Indias D.T y C. Septiembre de 2005

Señores:
Comité Evaluador
Departamento de Ingeniería Eléctrica y Electrónica

Apreciados señores:

Por medio de la presente me permito informarles que el trabajo de grado titulado "**SISTEMA DE MEDICION DE VARIABLES ELECTRICAS DE UN GENERADOR SINCRONO USANDO TELEMETRIA"** ha sido desarrollado de acuerdo a los objetivos establecidos.

Como director del proyecto consideramos que el trabajo es satisfactorio y amerita ser presentado para su evaluación.

Atentamente,

_____

M.Sc. EDUARDO GÓMEZ VÁSQUEZ

Nota de aceptación

_____

_____

_____

_____

Jurado

_____

Jurado

Cartagena de Indias D.T y C, Septiembre de 2005

AUTORIZACIÓN

Cartagena D.T.y C.

Yo Delder Alberto Arzuza Torres, identificado con cedula de ciudadania numero 73.193.631 de Cartagena, autorizo a la Universidad Tecnológica de Bolívar para hacer uso de mi trabajo de grado y publicarlo en el catalogo online de la Biblioteca.

_____

DELDER ALBERTO ARZUZA TORRES

AUTORIZACIÓN

Cartagena D.T.y C.

Yo José Ignacio Villarreal Marimón, identificado con cedula de ciudadania numero 73.196.475 de Cartagena, autorizo a la Universidad Tecnológica de Bolívar para hacer uso de mi trabajo de grado y publicarlo en el catalogo online de la Biblioteca.

_____

JOSÉ IGNACIO VILLARREAL MARIMÓN

ARTICULO 107

La institución se reserva el derecho de propiedad intelectual de todos los Trabajos de Grado aprobados, los cuales no pueden ser explorados comercialmente sin su autorización.

CONTENIDO

PAG.

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

LISTA DE ANEXOS

Pág.

INTRODUCCIÓN

Literalmente TELEMETRIA significa "medición a distancia" y es una tecnología que consiste en la medición de ciertas variables (velocidad, temperatura, aceleración, presión, voltaje, corriente, frecuencia...) mediante unos sensores (dispositivos que realizan la labor de SENSING, así llamaremos a nuestro sistema) y en envío de esos datos, a un centro de recepción, para su análisis.

Las comunicaciones inalámbricas están presentes en muchas de nuestras actividades diarias y su uso ha llegado a ser tan común, que perdemos la percepción de lo útil y a veces indispensable que pueden llegar a ser. Las redes celulares para transmitir voz y datos han surgido para proveer la movilidad y disponibilidad de la comunicación que el ritmo acelerado de vida de las grandes urbes exige. La utilización de sensores infrarrojos y de radiofrecuencia proveen la comodidad de controlar y operar a distancia aparatos electrónicos volviendo más sencillo nuestro quehacer diario. Asimismo, la creación de estándares de comunicaciones inalámbricas en las redes de transmisión de datos ha abierto oportunidades de desarrollo de estas tecnologías, aprovechando la utilización de interfaces aéreas operadas bajo frecuencias no licenciadas.

De cierta manera la Telemetría, podríamos definir en forma simple, como "la posibilidad de medir a distancia"[1] y más específicamente, como "la capacidad para leer datos remotos" mediante un sistema de

[1] Página de MovilGate http://www.chilemovil.com/cm/telemetry/faqs.html

comunicaciones con el propósito de realizar labores de SENSING sobre el dispositivo remoto.

En las redes inalámbricas el ancho de banda esta sujeto a variables como: la interferencia, el ruido en el ambiente, la movilidad de los dispositivos que se interconectan, etc. Los anteriores elementos hacen que el ancho de banda se adapte a estas variaciones, momento en el que comienzan a surgir elementos a los que se debe dar prioridad, se debe decidir cuales son las variables críticas a transmitir por el sistema de telemetría y en que orden, de acuerdo con las condiciones específicas actuales, en nuestro sistema de SENSING, las variables a medir son los voltajes y corrientes del generador sincrono.

Se requiere entonces, un buen diseño en el sistema de Telemetría que facilite las labores de medición realizadas sobre el generador sincrono, el diseño debe cumplir características de rendimiento como la seguridad en la transmisión de los datos (tanto hacia como desde el generador), mediante el uso de tecnología de redes inalámbricas.

PROLOGO

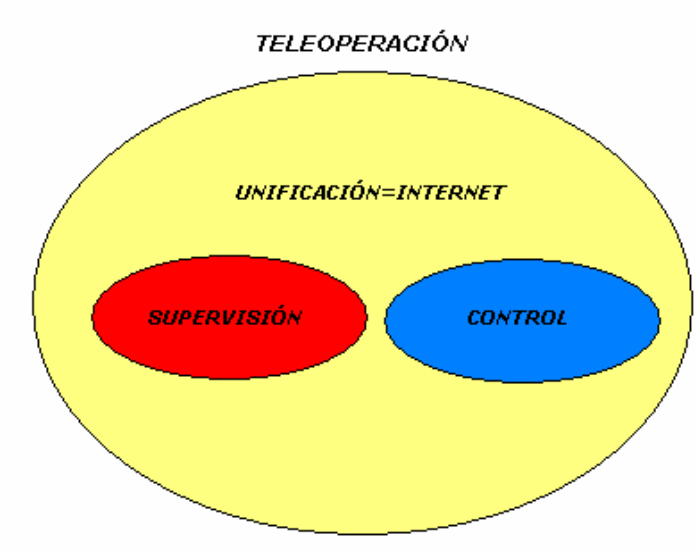
Hace años en sectores como la industria, las dos acciones importantes que son la supervisión y el control de las maquinas y equipos se hacía directamente sobre los elementos que se operaban. Luego con el avanzar y desarrollo de la tecnología se llegó a centros de operación, lugares que proveen de computadores que procesan y almacenan las señales y a estas ultimas se les muestran los resultados en tiempo real. Todo se visualiza en los llamados "mímicos", los cuales son mostrados en monitores.

La tecnología de hoy, trae consigo herramientas como el Internet de muy fácil acceso. Este ultimo se está utilizando, para que ya no importando en el lugar que se esté en el mundo, el operario e incluso el gerente pueda supervisar y operar las diferentes maquinas y equipos que componen los procesos de una planta. A todo este se le conoce con el nombre de tele operación.

La tele operación se divide en dos partes: la supervisión y el control, como se muestra en la figura 1, todo esto se unifica utilizando una tecnología, como Internet. La supervisión se encarga de la medición de variables asociadas a cada proceso en particular como corriente, voltaje, frecuencia, presión, etc. Además con la supervisión, se almacena la información en tiempo real, de forma periódica y automática o a solicitud del operador. Los principales elementos que lo componen son los equipos informáticos conectados en red local (servidor, pantallas gráficas e impresoras). Con el control se puede manipular las maquinas

y los equipos que hacen parte de cualquier proceso que se este llevando a cabo en una industria.

Figura 1. Diagrama de la tele operación



El macroproyecto que se visiona es la tele operación a un generador sincrono (el cual se encuentra en el laboratorio de maquinas de la UTB).

El alcance del presente trabajo de grado esta enmarcado en la supervisión (uno de los dos elementos que componen la tele operación y que corresponde a la figura 1, el área de color rojo) de las variables eléctricas asociadas a el generador sincrono en mención, como lo son la corriente y la tensión de cada una de sus fases. Para ello se tuvo que diseñar y validar (a través de pruebas de campo) del sistema de telemetría implementando una red inalámbrica; esta ultima esta conformada por el generador sincrono, un modulo de adquisición de datos, un transmisor y receptor inalámbrico y la interfaz grafica interactiva con el usuario.

Lo que proponemos es entonces, que se siga trabajando en los otros elementos importantes de la tele operación como lo son el de Control, para llegar al último peldaño que es la unificación.

1 MARCO TEORICO

Dentro del enorme horizonte de las comunicaciones inalámbricas y la computación móvil, las redes inalámbricas van ganando adeptos como una tecnología madura y robusta que permite resolver varios de los inconvenientes del uso del cable como medio físico de enlace en las comunicaciones, muchas de ellas de vital importancia en el trabajo cotidiano. Se introducen algunos conceptos relacionados con las redes locales inalámbricas WLAN y adelanta la futura competitividad de las diversas tecnologías IEEE 802.11, Bluetooth y HomeRF. Una vez que se ha tenido la oportunidad de haber hecho uso de algún dispositivo inalámbrico que proporcionase datos o información requerida con independencia del lugar, es prácticamente imposible olvidar las características que los hacen tan especiales. Muchos de los equipos inalámbricos otorgan la libertad necesaria para trabajar prácticamente desde cualquier punto del planeta e, incluso, permiten el acceso a todo tipo de información cuando se está de viaje. No importa que el sistema inalámbrico esté accediendo al correo electrónico desde un aeropuerto o recibiendo instrucciones desde el despacho para realizar alguna tarea, lo realmente relevante de esta tecnología es la extremada efectividad que se logra al poder mantener una conexión de datos con una red desde cualquier remoto sitio del globo mundial. Por otra parte, las comunicaciones de radio han estado a nuestra disposición desde hace ya bastante tiempo, teniendo como principal aplicación la comunicación mediante el uso de la voz. Hoy en día, millones de personas utilizan los sistemas de radio de dos vías para comunicaciones de voz punto a punto o multipunto. Sin embargo, aunque los ingenieros ya conocían las técnicas para modular una señal de radio con la cual conseguir el envío

de datos binarios, sólo recientemente han podido desarrollar y desplegar servicios de datos a gran escala.

Como muestra del complejo pero apasionante campo de las redes sin cables, el mundo de los denominados datos inalámbricos incluyen enlaces fijos de microondas, redes LAN inalámbricas, datos sobre redes celulares, redes WAN inalámbricas, enlaces mediante satélites, redes de transmisión digital, redes con paginación de una y dos vías, rayos infrarrojos difusos, comunicaciones basadas en láser, Sistema de Posicionamiento Global (GPS) y mucho más. Como se puede ver, una variada y extensa gama de tecnologías, muchas de las cuales son utilizadas con suma profusión por millones de usuarios en el transcurrir del día a día, sin saber cómo ni por qué la información ha llegado hasta ellos. Tampoco hay que olvidar los numerosos beneficios que aporta la utilización de los dispositivos inalámbricos. Ya que gracias a ellos se logran realizar conexiones imposibles para otro tipo de medio, conexiones a un menor costo en muchos escenarios, conexiones más rápidas, redes que son más fáciles y rápidas de instalar y conexiones de datos para usuarios móviles. Como vemos, el panorama de las redes inalámbricas es casi tan extenso o más que el de las propias redes convencionales, a las que estamos más habituados. Debido a la impresionante variedad de tecnologías, configuraciones, dispositivos, topologías y medios, relacionados con las redes inalámbricas debemos, muy a nuestro pesar, limitar la profundidad y extensión de este artículo centrándonos en las redes inalámbricas de área local. Este tipo de redes, por la proximidad al mundo de la pequeña y mediana empresa, las hace, ya no sólo mucho más asequibles, sino que su posible implantación en cualquier empresa o entorno de trabajo en grupo sea

una realidad totalmente tangible con la mera inversión de dichos medios, sin que los costes de adquisición sean el pesado lastre que impida el despegue definitivo de las redes inalámbricas. En síntesis, las redes LAN sin cables o más conocidas por el sobrenombre de WLAN (Wireless Local Area Network) no son algo realmente novedoso ni revolucionario dentro del mundo de la informática. Desde hace unos cuantos años, el atractivo de esta clase de redes hizo que aparecieran los primeros sistemas que utilizaban ondas de radio para interconectar computadores. El fundamento de muchas de las actuales redes inalámbricas se encuentra basado en el estándar IEEE 802.11, y más concretamente en las especificaciones IEEE 802.11b, IEEE 802.11a y IEEE 802.11g. Un consorcio, el "Wireless Ethernet Compatibility Alliance" (WECA), formado por un nutrido grupo de relevantes empresas, ha creado una nueva línea de productos de mayores prestaciones y de plena compatibilidad.
Con este estándar se pone fin a la larga tradición que siempre ha acompañando y se ha relacionado con el mundo de las redes inalámbricas.

Además, los productos acogidos a la normativa IEEE 802.11 tienen garantizada la interoperatividad entre fabricantes, consiguiendo al mismo tiempo una significativa reducción de los costes y abaratamiento de los dispositivos para el usuario final.

Este consorcio ha establecido un estándar llamado Wi-Fi que permite certificación de los productos acogidos a esta normativa para lograr que entre ellos existan una obligada interoperatividad y otros aspectos comunes de actuación como la facilidad de configuración, unanimidad de

protocolos, modos de funcionamiento, así como las más elementales normas. Pero, independientemente del esperanzador futuro de las WLAN acogidas al Wi-Fi, dentro de este particular sector de las redes inalámbricas hay otras tecnologías que también aprovechan parte de la infraestructura de la cual hacen uso casi todos los dispositivos WLAN. En general, los sistemas LAN sin cables basados en el protocolo 802.11 hacen un exhaustivo uso de la banda de frecuencias de los 2,4 GHz. El porqué de este concreto rango de frecuencias no es difícil de explicar y puede resumirse en que en esta zona del espectro electromagnético no se requiere el uso de licencias tal y como se lleva a cabo la regulación de los sistemas de radio, ya que en ellas se permite la transmisión de información en bandas del espectro, concretamente en las bandas llamadas ISM por su uso para aplicaciones industriales, científicas y médicas (ISM Industrial scientific medical). Pero esta misma ventaja actúa a su vez de atractivo y poderoso reclamo para otras tecnologías, sistemas o dispositivos inalámbricos que también quieran basar su funcionamiento en esta área específica del espectro.

Las WLAN aunque son la base de la expansión y flexibilidad de muchas de las actuales redes LAN, pecan quizá de ser una solución más bien general y dirigida a entornos de trabajo en grupo y empresas que puedan sacar el máximo partido a sus capacidades. Precisamente, esta generalidad ha dado pie a que nuevas tecnologías como Bluetooth y HomeRF, surjan en torno al protocolo 802.11, y aprovechando igualmente el rango de frecuencias de 2,4GHz han optado por especializarse en ofrecer una conectividad inalámbrica, por supuesto, pero enfocada a unos usos mucho más particulares y en relación directa con los futuros hábitos de vida de los componentes de la moderna,

activa y tecnológicamente sofisticada sociedad de principios del siglo XXI.

Bluetooth es, a grandes rasgos, una especificación para la industria informática y de las telecomunicaciones que describe un método de conectividad móvil universal con el cual se pueden interconectar dispositivos como teléfonos móviles, Asistentes Personales Digitales (PDA), ordenadores y muchos otros dispositivos, ya sea en el hogar, en la oficina o, incluso, en el automóvil, utilizando una conexión inalámbrica de corto alcance. Es un estándar que describe la manera en la que una enorme variedad de dispositivos pueden conectarse entre sí, de una forma sencilla y sincronizada, con cualquier otro equipo que soporte dicha tecnología utilizando las ondas de radio como medio de transporte de la información. Técnicamente, la implementación de esta novedosa tecnología no entraña ninguna complicación técnica especialmente problemática ni sofisticada. Tampoco supone que los nuevos dispositivos equipados con esta tecnología deban sufrir profundas revisiones o modificaciones, todo lo contrario. En sí, cada dispositivo deberá estar equipado con un pequeño chip que transmite y recibe información a una velocidad de 1Mbps en la banda de frecuencias de 2,4 GHz que está disponible en todo el mundo, con ciertas particularidades según los diferentes países de aplicación, ya que es empleada con enorme profusión en numerosos dispositivos.

Con una finalidad muy similar, la tecnología HomeRF, basada en el protocolo de acceso compartido ("Shared Wireless Access Protocol" - SWAP), encamina sus pasos hacia la conectividad sin cables dentro del hogar. Los principales seguidores de estos sistemas, se agrupan en

torno al Consorcio que lleva su mismo nombre HomeRF, teniendo a Proxim (una filial de Intel) como el miembro que más empeño esta realizando en la implantación de dicho estándar. Además de la sombra de Intel, Compaq es otra de las firmas relevantes que apoya el desarrollo de producto HomeRF. El soporte a esta tecnología se materializa en que actualmente ambas significativas firmas poseen cada una de ellas un producto bajo esta novedosa configuración. Al igual que WECA o Bluetooth SIG ("Bluetooth Special Interest Group"), el HomeRF Working Group (HRFWG) es un grupo de compañías encargadas de proporcionar y establecer un cierto orden en este océano tecnológico, obligando que los productos fabricados por las empresas integrantes de este grupo tengan una buena interoperatividad. Por si toda esta competitividad no fuera suficiente, el Instituto de Estándares de Telecomunicaciones Europeo (ETSI) es otra de las reconocidas organizaciones de estandarización, responsable, entre otros, de haber desarrollado el estándar GSM para la telefonía celular digital. También son responsables de haber llevado a cabo durante los años 1991 y 1996 el proyecto HyperLAN, en el cual su objetivo primordial estaba conseguir una tasa de transferencia mayor que la ofrecida por la especificación IEEE 802.11. Según los estudios realizados, HyperLAN incluía cuatro estándares diferentes, de los cuales el denominado Tipo 1, es el que verdaderamente se ajusta a las necesidades futuras de las WLAN, estimándose una velocidad de transmisión de 23,5 Mbps, notablemente superior a los 1 ó 2 Mbps de la normativa IEEE 802.11b. Actualmente, el ETSI dispone de la especificación LANHiper2 que mejora notablemente las características de sus antecesoras, ofreciendo una mayor velocidad de transmisión en la capa física de 54 Mbps para lo cual emplea el método de modulación OFDM (Orthogonal Frequency Digital

Multiplexing) y ofrece soporte QoS. Bajo esta especificación se ha formado un grupo de reconocidas firmas el HiperLAN2 Global Forum (H2GF), con la intención de sacar al mercado productos basados en ese competitivo estándar.

El medio cambiante y las condiciones en las que operan las redes inalámbricas hacen que aún sean consideradas como medio alternativo para ofrecer servicios. La inseguridad inherente en este tipo de redes debido a la facilidad de acceso al medio de transmisión las hace muy vulnerables a ataques frecuentes. Son dos de las principales desventajas de las redes inalámbricas.

Campos en los que se está investigando profundamente con el fin de superarlas y convertirlas en un medio importante de transmisión de datos. La calidad en la transmisión de los datos requiere de una adaptación particular de este tipo de redes a este medio cambiante. En forma simple, la calidad en el servicio en redes inalámbricas se puede ofrecer ya sea a través de la reservación de recursos en la red o a través de la adaptación de la aplicación que se ejecuta sobre la red. Este trabajo de grado esta orientado a establecer un sistema de comunicación inalámbrico que provee la Telemetría entre un generador sincrono y la computadora donde se encuentra la interfaz grafica de usuario para su análisis.

2 METODOLOGÍA

Durante el proceso investigativo se hizo un estudio de la documentación actualizada sobre los temas de interés relacionados con las redes inalámbricas. Al mismo tiempo se revisó la disponibilidad de los equipos para el diseño del sistema, realizando una evaluación de la conveniencia de cada uno de ellos de acuerdo a factores como: costo, eficiencia en su operación, facilidad de configuración y otros.

Luego de adquirir estos elementos el trabajo restante correspondió a pruebas de laboratorio y de campo hasta que los resultados alcanzaron los objetivos definidos.

A continuación se presenta cada una de las etapas que se llevaron a cabo para el desarrollo del presente trabajo de grado.

2.1 SELECCIÓN DEL PROTOCOLO DE COMUNICACIONES INALÁMBRICAS

Inicialmente se consultó en los diferentes medios: Internet, libros, revistas especializadas, etc., los diferentes protocolos que se utilizan tanto a nivel investigativo y comercial, para establecer las comunicaciones inalámbricas, así como también los distintos tipos de terminales remotas de campo que cada estándar de comunicaciones utiliza, algunos de estos equipos se muestran en la figura 2. Se definieron las ventajas y desventajas de cada uno para así seleccionar la tecnología más adecuada para nuestro sistema de telemetría.

Figura 2. Distintos tipos de terminales remotas de campo

Fuente: Imágenes descargadas de la Web.

2.1.1 Comparación de los diferentes protocolos y elección final.

Después de una exhaustiva investigación se llegó a la conclusión que el estándar **Bluetooth** es la tecnología más adecuada para nuestro diseño puesto que cumple con características como son el bajo coste, tamaño reducido, bajo consumo de potencia, ya que este protocolo no necesita iniciarse: siempre esta trabajando en un segundo plano y algo muy importante que es que utiliza la banda libre de licencias llamada ISM (Industrial Scientific Medical), de sus siglas en ingles Industria, Científica y Medica. La cual es gratis y no necesita pedir licencia al Ministerio de Comunicaciones.

Esta tecnología se comparó con tecnologías similares y que se puede decir que son sus "rivales" más cercanos. Entre ellas tenemos a IrDA más conocido como infrarrojo, HomeRF y WLAN.

Tabla 1. Bluetooth Vs IrDa.

	Infrarrojos	Bluetooth
Rata de Transmisión (Datos)	4 Mbps	1 Mbps (próxima a 10Mbps)
línea de vista?	Si	No
Conectar mas de dos dispositivos?	No	Si (Hasta 8 dispositivos)
Internet	Si	Si
Capacidad de voz	No	Si

Fuente: Cardozo Carlos. Tecnología Bluetooth: conexión sin cables. Universidad Tecnológica de Bolívar. 2003

2.1.1.1 Bluetooth Vs IrDa.

Una de las diferencias entre Bluetooth e infrarrojos es que Bluetooth no requiere "línea de vista" para transferir datos. Además la tecnología Bluetooth esta habilitada para transferir voz y datos, esto permite que los usuarios tengan una red de área personal (PAN), definida en un radio de 10-100m. En este rango, los dispositivos Bluetooth pueden comunicarse entre si. En la Tabla 1 se muestran sus diferencias.

2.1.1.2 Bluetooth Vs HomeRF y WLAN.

HomeRF es posiblemente una tecnología de competencia para Bluetooth, sin embargo esto no puede asegurarse debido a que Bluetooth podría reemplazar a HomeRF e incluso podría coexistir con ella. Lo que si esta claro, es que HomeRF no puede sustituir a Bluetooth para ámbitos fuera del hogar. Una situación similar ocurre cuando un dispositivo Bluetooth ingresa a una oficina provista de una WLAN (Wírelles LAN, LAN Inalámbrica). Aunque Bluetooth, HomeRF y WLAN operan a la misma

frecuencia, la tasa de transferencia para estos dispositivos podría variar significativamente. Sin embargo, la coexistencia entre ellos es posible debido a una técnica llamada expansión de espectro, con la cual se minimizan las interferencias para dispositivos que trabajan a la misma frecuencia. En las tablas 2 y 3 se muestran sus principales diferencias.

Tabla 2. Bluetooth Vs HomeRF.

	HomeRF	**Bluetooth**
Ambiente	Hogar	Usuarios
Rango	50 mts	10 a 100 mts
Saltos de Frecuencia	50 Saltos / seg	1600 saltos / seg
estándar Abierto?	Si	Si
Frecuencia	2,4 Ghz(Banda ISM)	2,4 Ghz(Banda ISM)
Datos y voz	Dato y 6 canales de voz	Dato y 3 canales de voz
Rata de transmisión	1 Mbps	1 Mbps

Fuente: Cardozo Carlos. Tecnología Bluetooth: conexión sin cables. Universidad Tecnológica de Bolívar. 2003

Tabla 3. Bluetooth Vs WLAN.

	WLAN	**Bluetooth**
Tasa de transmisión	11 Mbps	1 Mbps
Rango	30 a 1000 mts	10 a 100 mts
Frecuencia	2,4 Ghz(Banda ISM)	2,4 Ghz(Banda ISM)

Fuente: Cardozo Carlos. Tecnología Bluetooth: conexión sin cables. Universidad Tecnológica de Bolívar. 2003

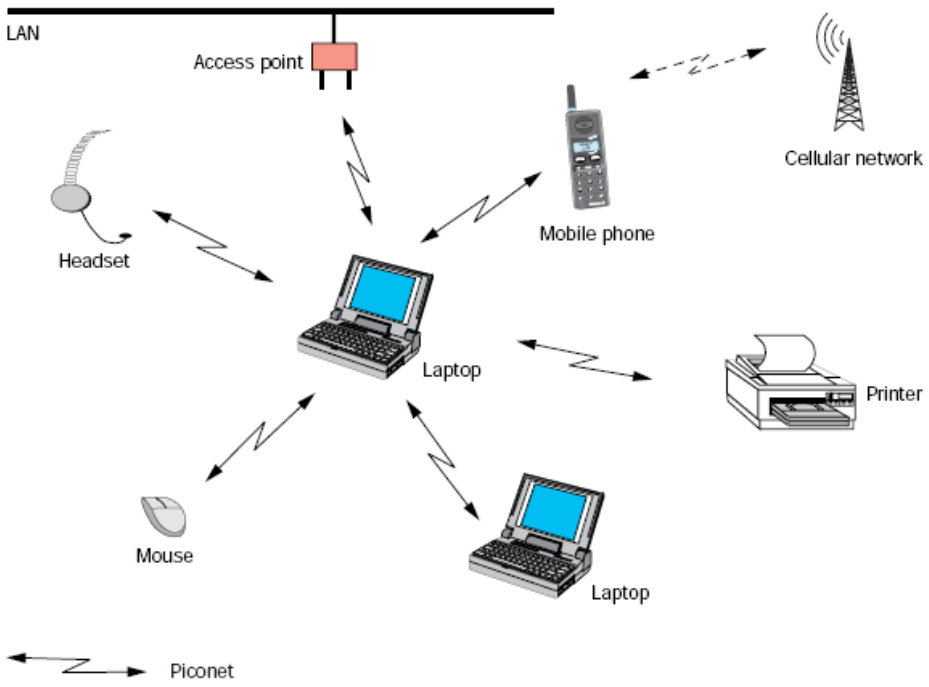
2.2 BÚSQUEDA Y ADQUISICIÓN DE INFORMACIÓN

Se buscó y se recopiló información concerniente a los diferentes elementos que componen nuestro sistema de telemetría.

2.2.1 Protocolo elegido para realizar la comunicación inalámbrica.

Lo primero que se hizo fue de investigar acerca de la tecnología escogida para realizar nuestro enlace inalámbrico. Se consultó su página Web oficial[2], donde se encuentra todo lo relacionado con los aspectos técnicos y sus especificaciones. También se hizo contacto con sus expertos a través del correo electrónico para solucionar nuestras dudas.

Figura 3. Modelo Usado para conectividad de área Personal.



Fuente: http://www.ericsson.com/about/publications/review/1998_03/files/1998031.pdf

[2] www.bluetooth.org también se puede consultar www.bluetooth.com

2.2.1.1 Que es Bluetooth[3].

Es la norma que define un Standard global de comunicación inalámbrica, que posibilita la transmisión de voz y datos entre diferentes equipos mediante un enlace por radiofrecuencia. Los principales objetivos que se pretende conseguir con esta norma son:

- Facilitar las comunicaciones entre equipos móviles y fijos
- Eliminar cables y conectores entre éstos.
- Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre nuestros equipos personales, como se muestra en la figura 3.

La tecnología Bluetooth comprende hardware, software y requerimientos de inter-operatibilidad, por lo que para su desarrollo ha sido necesaria la participación de los principales fabricantes de los sectores de las telecomunicaciones y la informática, tales como: **Ericsson, Nokia, Toshiba, IBM, Intel** y otros. Posteriormente se han ido incorporando muchas más compañías, y se prevé que próximamente los hagan también empresas de sectores tan variados como: automatización industrial, maquinaria, ocio y entretenimiento, fabricantes de juguetes, electrodomésticos, etc., con lo que en poco tiempo se nos presentará un panorama de total conectividad de nuestros aparatos tanto en casa como en el trabajo.

2.2.1.2 Antecedentes

En 1994 Ericsson inició un estudio para investigar la viabilidad de una interfase vía radio, de bajo coste y bajo consumo, para la interconexión entre teléfonos móviles y otros accesorios con la intención de eliminar

[3] http://www.ericsson.com/about/publications/review/1998_03/files/1998031.pdf

cables entre aparatos. El estudio partía de un largo proyecto que investigaba sobre unos multi-comunicadores conectados a una red celular, hasta que se llegó a un enlace de radio de corto alcance, llamado *MC link.* Conforme éste proyecto avanzaba se fue viendo claro que éste tipo de enlace podía ser utilizado ampliamente en un gran número de aplicaciones, ya que tenia como principal virtud el que se basaba en un chip de radio relativamente económico.

2.2.1.3 El SIG

A comienzos de 1997, según avanzaba el proyecto MC link, Ericsson fue despertando el interés de otros fabricantes de equipos portátiles. En seguida se vió claramente que para que el sistema tuviera éxito, un gran número de equipos deberían estar equipados con ésta tecnología. Esto fue lo que originó a principios de 1998, la creación de un grupo de interés especial (**SIG**), formado por 5 promotores que fueron: Ericsson, Nokia, IBM, Toshiba e Intel. La idea era lograr un conjunto adecuado de áreas de negocio, dos líderes del mercado de las telecomunicaciones, dos líderes del mercado de los PCS portátiles y un líder de la fabricación de chips. El propósito principal del consorcio fué y es, el establecer un standard para la *interface* aérea junto con su software de control, con el fin de asegurar la interoperatibilidad de los equipos entre los diversos fabricantes.

2.2.1.4 La interfase aérea Bluetooth

El primer objetivo para los productos Bluetooth de primera generación eran los entornos de la gente de negocios que viaja frecuentemente. Por lo que se debería pensar en integrar el chip de radio Bluetooth en equipos como: PCS portátiles, teléfonos móviles, PDAs y auriculares.

Esto originaba una serie de cuestiones previas que deberían solucionarse tales como:

- El sistema debería operar en todo el mundo.
- El emisor de radio deberá consumir poca energía, ya que debe integrarse en equipos alimentados por baterías.
- La conexión deberá soportar voz y datos, y por lo tanto aplicaciones multimedia.

2.2.1.5 Banda de frecuencia libre

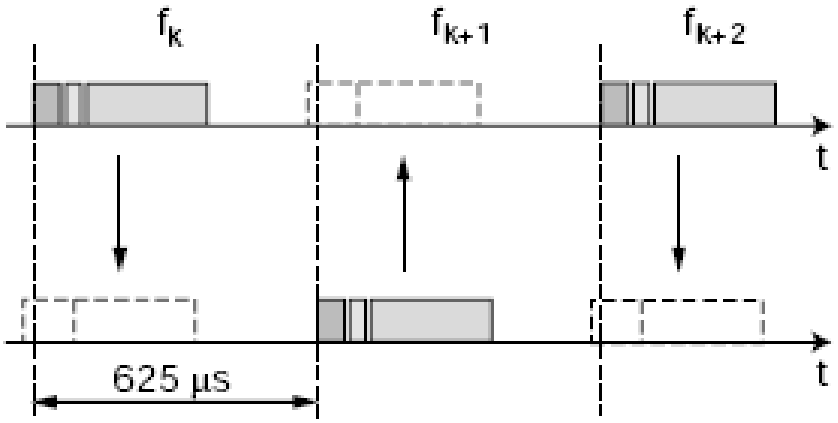
Para poder operar en todo el mundo es necesaria una banda de frecuencia abierta a cualquier sistema de radio independientemente del lugar del planeta donde nos encontremos. Sólo la banda ISM (médico-científica internacional) de 2,45 Ghz cumple con éste requisito, con rangos que van de los 2.400 Mhz a los 2.500 Mhz, y solo con algunas restricciones en países como Francia, España y Japón.

2.2.1.6 Salto de frecuencia

Debido a que la banda ISM está abierta a cualquiera, el sistema de radio Bluetooth deberá estar preparado para evitar las múltiples interferencias que se pudieran producir. Éstas pueden ser evitadas utilizando un sistema que busque una parte no utilizada del espectro o un sistema de salto de frecuencia. En los sistemas de radio Bluetooth se suele utilizar el método de salto de frecuencia debido a que ésta tecnología puede ser integrada en equipos de baja potencia y bajo coste. Éste sistema divide la banda de frecuencia en varios canales de salto, donde, los transceptores, durante la conexión van cambiando de uno a otro canal de salto de manera pseudo-aleatoria. Con esto se consigue que el ancho de banda instantáneo sea muy pequeño y también una propagación

efectiva sobre el total de ancho de banda. En conclusión, con el sistema FH (Salto de frecuencia), se pueden conseguir transceptores de banda estrecha con una gran inmunidad a las interferencias.

Figura 4. Salto de frecuencia/división de tiempo duplex.



Fuente: http://www.ericsson.com/about/publications/review/1998_03/files/1998031.pdf

2.2.1.7 Definición de canal.

Como hemos comentado, Bluetooth utiliza un sistema FH/TDD (salto de frecuencia/división de tiempo duplex), como se muestra en la figura 4 en el que el canal queda dividido en intervalos de 625 µs, llamados slots, donde cada salto de frecuencia es ocupado por un slot. Esto da lugar a una frecuencia de salto de 1600 veces por segundo, en la que un paquete de datos ocupa un slot para la emisión y otro para la recepción y que pueden ser usados alternativamente, dando lugar a un esquema de tipo TDD.

Dos o más unidades Bluetooth pueden compartir el mismo canal dentro de una **piconet** , donde una unidad actúa como maestra, controlando el tráfico de datos en la piconet que se genera entre las demás unidades, donde estas actúan como esclavas, enviando y recibiendo señales hacia

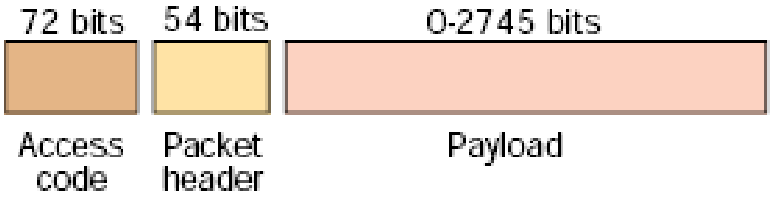
el maestro. El salto de frecuencia del canal está determinado por la secuencia de la señal, es decir, el orden en que llegan los saltos y por la fase de ésta secuencia. En Bluetooth, la secuencia queda fijada por la identidad de la unidad maestra de la piconet (un código único para cada equipo), y por su frecuencia de reloj. Por lo que, para que una unidad esclava pueda sincronizarse con una unidad maestra, ésta primera debe añadir un ajuste a su propio reloj nativo y así poder compartir la misma portadora de salto.

En países donde la banda está abierta a 80 canales o más, espaciados todos ellos a 1 Mhz., se han definido 79 saltos de portadora, y en aquellos donde la banda es más estrecha se han definido 23 saltos.

2.2.1.8 Definición de paquete.

La información que se intercambia entre dos unidades Bluetooth se realiza mediante un conjunto de slots que forman un paquete de datos (figura 5). Cada paquete comienza con un código de acceso de 72 bits, que se deriva de la identidad maestra, seguido de un paquete de datos de cabecera de 54 bits. Éste contiene importante información de control, como tres bits de acceso de dirección, tipo de paquete, bits de control de flujo, bits para la retransmisión automática de la pregunta, y chequeo de errores de campos de cabeza. Finalmente, el paquete que contiene la información, que puede seguir al de cabeza, tiene una longitud de 0 a 2745 bits. En cualquier caso, cada paquete que se intercambia en el canal está precedido por el código de acceso.

Figura 5. Paquete de formato fijo.



Fuente: www.zonabluetooth.com

Los receptores de la piconet comparan las señales que reciben con el código de acceso, si éstas no coinciden, el paquete recibido no es considerado como válido en el canal y el resto de su contenido es ignorado.

2.2.1.9 Definición de enlace físico.

En la especificación Bluetooth se han definido dos tipos de enlace que permitan soportarr incluso aplicaciones multimedia:

- Enlace de sincronización de conexión orientada (SCO)
- Enlace asíncrono de baja conexión (ACL)

Los enlaces SCO soportan conexiones asimétricas, punto a punto, usadas normalmente en conexiones de voz, éstos enlaces están definidos en el canal, reservándose dos slots consecutivos (envío y retorno) en intervalos fijos. Los enlaces ACL soportan conmutaciones punto a punto simétricas o asimétricas, típicamente usadas en la transmisión de datos.

Un conjunto de paquetes se han definido para cada tipo de enlace físico:

- Para los enlaces SCO, existen tres tipos de slot simple, cada uno con una portadora a una velocidad de 64 kbit/s. La transmisión de

voz se realiza sin ningún mecanismo de protección, pero si el intervalo de las señales en el enlace SCO disminuye, se puede seleccionar una velocidad de corrección de envio de 1/3 o 2/3.

- Para los enlaces ACL, se han definido el slot-1, slot-3, slot-5. Cualquiera de los datos pueden ser enviados protegidos o sin proteger con una velocidad de corrección de 2/3. La máxima velocidad de envío es de 721 kbit/s en una dirección y 57.6 kbit/s en la otra.

2.2.1.10 Inmunidad a las interferencias

Como se mencionó anteriormente Bluetooth opera en una banda de frecuencia que está sujeta a considerables interferencias, por lo que el sistema ha sido optimizado para evitar éstas interferencias. En este caso La técnica de salto de frecuencia es aplicada a una alta velocidad y una corta longitud de los paquetes (1600 saltos/segundo, para slots-simples). Los paquetes de datos están protegido por un esquema ARQ (repetición automática de consulta), en el cual los paquetes perdidos son automáticamente retransmitidos, aun así, con este sistema, si un paquete de datos no llegase a su destino, sólo una pequeña parte de la información se perdería. La voz no se retransmite nunca, sin embargo, se utiliza un esquema de codificación muy robusto. Éste esquema, que está basado en una modulación variable de declive delta (CSVD), que sigue la forma de la onda de audio y es muy resistente a los errores de bits. Éstos errores son percibidos como ruido de fondo, que se intensifica si los errores aumentan.

2.2.1.11 Piconets

Si un equipo se encuentra dentro del radio de cobertura de otro, éstos pueden establecer conexión entre ellos. En principio sólo son necesarias un par de unidades con las mismas características de hardware para establecer un enlace. Dos o más unidades Bluetooth que comparten un mismo canal forman una piconet. Para regular el tráfico en el canal, una de las unidades participantes se convertirá en maestra, pero por definición, la unidad que establece la piconet asume éste papel y todos los demás serán esclavos. Los participantes podrían intercambiar los papeles si una unidad esclava quisiera asumir el papel de maestra. Sin embargo sólo puede haber un maestro en la piconet al mismo tiempo.

Cada unidad de la piconet utiliza su identidad maestra y reloj nativo para seguir en el canal de salto. Cuando se establece la conexión, se añade un ajuste de reloj a la própia frecuencia de reloj nativa de la unidad esclava para poder sincronizarse con el reloj nativo del maestro. El reloj nativo mantiene siempre constante su frecuencia, sin embargo los ajustes producidos por las unidades esclavas para sincronizarse con el maestro, sólo son válidos mientras dura la conexión.

Como ya hemos comentado, las unidades maestras controlan el tráfico del canal, por lo que estas tienen la capacidad para reservar slots en los enlaces **SCO**. Para los enlaces **ACL**, se utiliza un esquema de sondeo. A una esclava sólo se le permite enviar un slot a un maestro cuando ésta se ha dirigido por su dirección MAC (medio de control de acceso) en el procedimiento de slot maestro-esclavo. Éste tipo de slot implica un sondeo por parte del esclavo, por lo que, en un tráfico normal de paquetes, este es enviado a una urna del esclavo automáticamente. Si la información del esclavo no está disponible, el maestro puede utilizar

un paquete de sondeo para sondear al esclavo explícitamente. Los paquetes de sondeo consisten únicamente en uno de acceso y otro de cabecera. Éste esquema de sondeo central elimina las colisiones entre las transmisiones de los esclavos.

2.2.1.12 Estableciendo conexión.

De un conjunto total de 79 (23) portadoras del salto, un subconjunto de 32(16) portadoras activas han sido definidas. El subconjunto, que es seleccionado pseudo-aleatóriamente, se define por una única identidad.

Acerca de la secuencia de activación de las portadoras, se establece que, cada una de ellas visitará cada salto de portadora una sola vez, con una longitud de la secuencia de 32 (16) saltos. En cada uno de los 2.048 (1.028) saltos, las unidades que se encuentran en modon *standby* (en espera) mueven sus saltos de portadora siguiendo la secuencia de las unidades activas. El reloj de la unidad activa siempre determina la secuencia de activación.

Durante la recepción de los intervalos, en los últimos 18 slots o 11,25 ms, las unidades escuchan una simple portadora de salto de activación y correlacionan las señales entrantes con el código de acceso derivado de su propia identidad. Si los triggers son correlativos, esto es, si la mayoría de los bits recibidos coinciden con el código de acceso, la unidad se auto-activa e invoca un procedimiento de ajuste de conexión. Sin embargo si estas señales no coinciden, la unidad vuelve al estado de reposo hasta el siguiente evento activo.

Para establecer la piconet, la unidad maestra debe conocer la identidad del resto de unidades que están en modo standby en su radio de

cobertura. El maestro o aquella unidad que inicia la piconet transmite el código de acceso continuamente en periodos de 10 ms, que son recibidas por el resto de unidades que se encuentran en *standby.* El tren de 10 ms. de códigos de acceso de diferentes saltos de portadora, se transmite repetidamente hasta que el receptor responde o bien se excede el tiempo de respuesta.

Cuando una unidad emisora y una receptora seleccionan la misma portadora de salto, la receptora recibe el código de acceso y devuelve una confirmación de recibo de la señal, es entonces cuando la unidad emisora envía un paquete de datos que contiene su identidad y frecuencia de reloj actual. Después de que el receptor acepta éste paquete, ajustará su reloj para seleccionar el canal de salto correcto determinado por emisor. De éste modo se establece una piconet en la que la unidad emisora actúa como maestra y la receptora como esclava. Después de haber recibido los paquetes de datos con los códigos de acceso, la unidad maestra debe esperar un procedimiento de requerimiento por parte de las esclavas, diferente al proceso de activación, para poder seleccionar una unidad específica con la que comunicarse.

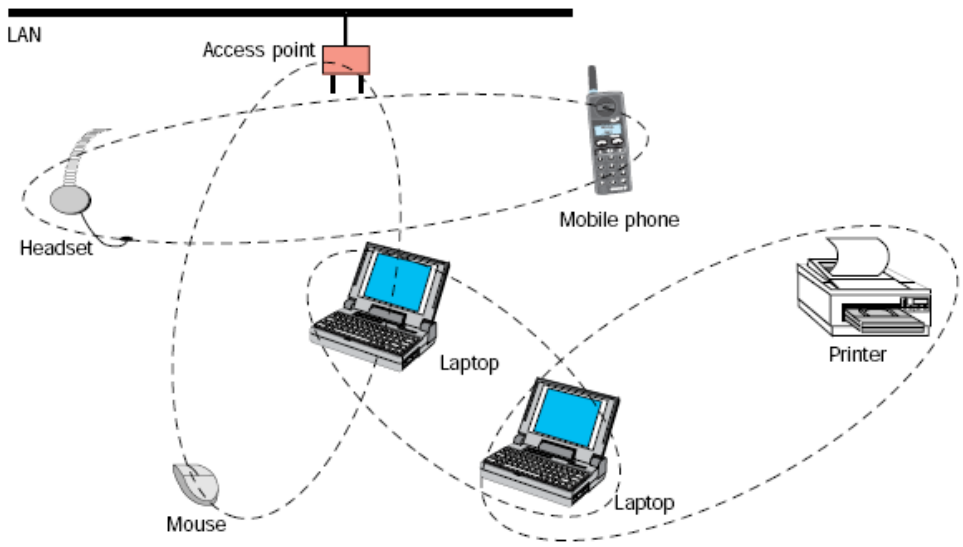
El número máximo de unidades que pueden participar activamente en una simple piconet es de 8, un maestro y siete esclavos, por lo que la dirección MCA del paquete de cabecera que se utiliza para distinguir a cada unidad dentro de la piconet, se limita a tres bits.

2.2.1.13 Scatternet

Los equipos que comparten un mismo canal sólo pueden utilizar una parte de su capacidad de este. Aunque los canales tienen un ancho de

banda de un 1Mhz, cuantos más usuarios se incorporan a la piconet, disminuye la capacidad hasta unos 10 kbit/s más o menos. Teniendo en cuenta que el ancho de banda medio disponible es de unos 80 Mhz en Europa y USA (excepto en España y Francia), éste no puede ser utilizado eficazmente, cuando cada unidad ocupa una parte del mismo canal de salto de 1Mhz. Para poder solucionar éste problema se adoptó una solución de la que nace el concepto de scatternet, Como se muestra en la figura 6.

Figura 6. Una scatternet de cuatro piconets aplicada al escenario de la figura 3.

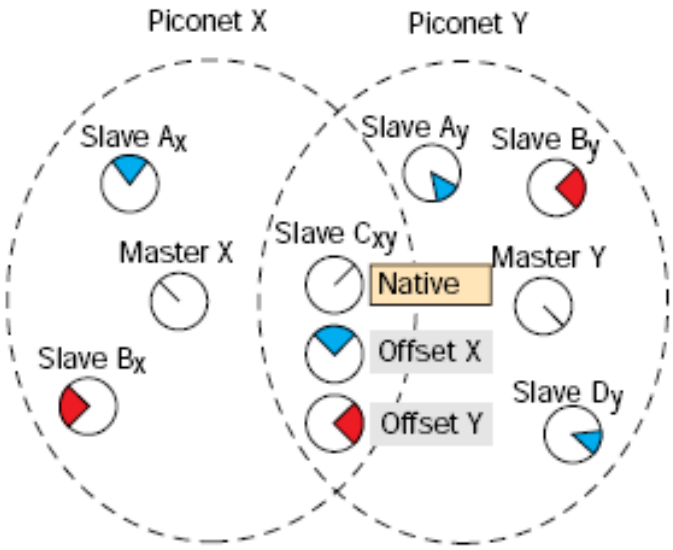


Fuente: http://www.ericsson.com/about/publications/review/1998_03/files/1998031.pdf

Las unidades que se encuentran en el mismo radio de cobertura pueden establecer potencialmente comunicaciones entre ellas. Sin embargo, sólo aquellas unidades que realmente quieran intercambiar información comparten un mismo canal creando la piconet, como por ejemplo en la figura 7 el esclavo Cxy, pertenece a dos piconets al mismo tiempo. Éste

hecho permite que se creen varias piconets en áreas de cobertura superpuestas. A un grupo de piconets se le llama scatternet. El rendimiento, en conjunto e individualmente de los usuarios de una scatternet es mayor que el que tiene cada usuario cuando participa en un mismo canal de 1 Mhz. Además, estadísticamente se obtienen ganancias por multiplexión y rechazo de canales salto. Debido a que individualmente cada piconet tiene un salto de frecuencia diferente, diferentes piconets pueden usar simultáneamente diferentes canales de salto.

Figure 7. Participación del esclavo Cxy en dos piconets.



Fuente: http://www.ericsson.com/about/publications/review/1998_03/files/1998031.pdf

Hemos de tener en cuenta que cuantas más piconets se añaden a la scatternet el rendimiento del sistema **FH** disminuye poco a poco, habiendo una reducción por termino medio del 10%. sin embargo el rendimiento que finalmente se obtiene de múltiples piconets supera al de una simple piconet.

2.2.1.14 Seguridad

Para asegurar la protección de la información se ha definido un nivel básico de encriptación, que se ha incluido en el diseño del clip de radio para proveer de seguridad en equipos que carezcan de capacidad de procesamiento, las principales medidas de seguridad son:

- Una rutina de pregunta-respuesta, para autentificación
- Una corriente cifrada de datos, para encriptación
- Generación de una clave de sesión (que puede ser cambiada durante la conexión)

Tres entidades son utilizadas en los algoritmos de seguridad: la dirección de la unidad Bluetooth, que es una entidad pública; una clave de usuario privada, como una entidad secreta; y un número aleatorio, que es diferente por cada nueva transacción.

Como se ha descrito anteriormente, la dirección Bluetooth se puede obtener a través de un procedimiento de consulta. La clave privada se deriva durante la inicialización y no es revelada posteriormente. El número aleatorio se genera en un proceso pseudo-aleatorio en cada unidad Bluetooth.

Con toda esta información se llegó al punto de buscar adaptadores de puerto bluetooth que cumplieran las exigencias de nuestro sistema de telemetría tales como facilidad de configuración e instalación, alcance, documentación y por que no, el precio.

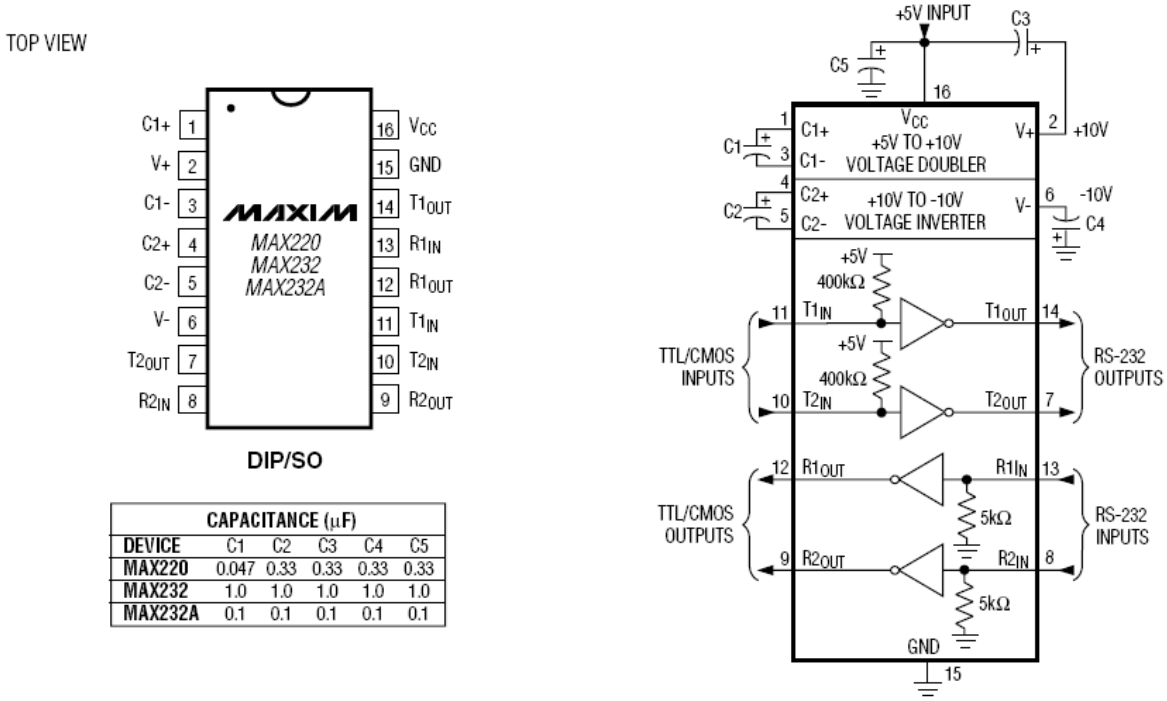
2.2.2 Microcontrolador

Se consultó en Internet cuales son los microcontroladores que más se utilizan en este tipo de aplicaciones y se llegó a la conclusión que era el PIC 16F877, de la familia de MicroCHIPs[4].

Se tuvo que investigar como se programan estos microcontroladores como también ejemplos de programas y planos de las tarjetas que se utilizan para su comunicación. En el anexo A se encuentra sus configuraciones técnicas, tales como sus pines, tensión de entrada, oscilador a utilizar, etc.

2.2.3 Comunicación serial

Figura 8. Características técnicas del MAX232



CAPACITANCE (μF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	0.047	0.33	0.33	0.33	0.33
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

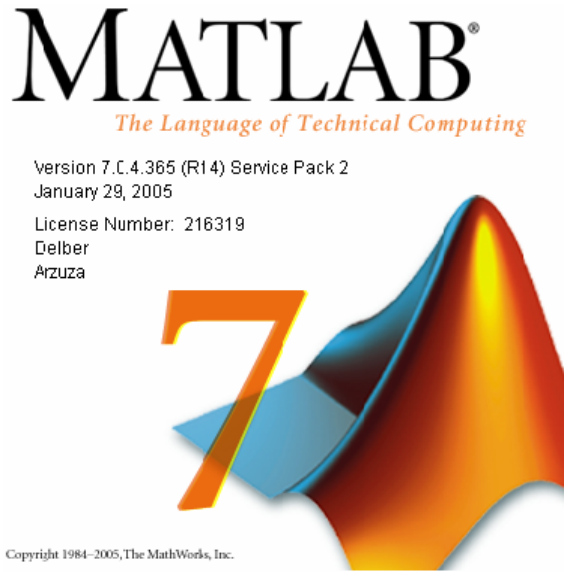
Fuente: Datasheet del MAX232

[4] www.microchips.com

Como el formato de las tramas de los microcontroladores son seriales se tuvo que investigar como se realiza esta comunicación utilizando la interfaz RS/232. Se investigaron entre otras sus características técnicas tales como conectores, cables, velocidad de transmisión, control de flujo, etc. Así como también se obtuvo información de cómo realizar este enlace con el microcontrolador, investigación que tuvo como resultado la implementación del MAX232. Las características técnicas de este integrado se muestran en la figura 8.

2.2.4 Matlab 7.0.4

Figura 9. Versión de MATLAB 7.0.4

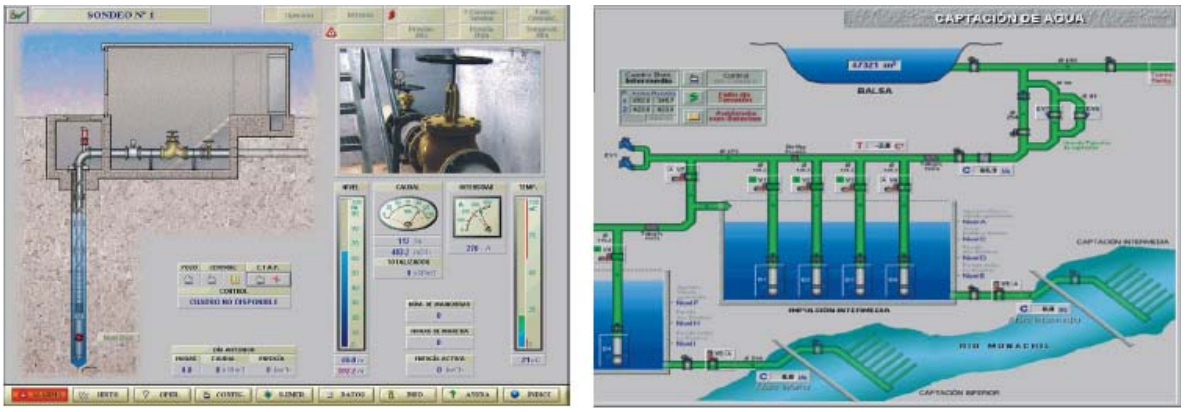


Fuente: Plataforma MATLAB 7.0.4.

Se escogió a MATLAB 7.0.4 (figura 9) como plataforma para realizar el software en que se desarrolló la interfaz grafica de usuario, y el procesamiento de datos que lleguen al computador. Esta plataforma tiene la ventaja de manejar el puerto serial, que es el que se utilizó para llevar a cabo nuestro sistema de comunicaciones. Se consultó

información sobre: programación, diseño de la interfaz grafica de usuario (en la figura 10 se presentan algunos ejemplos de software de supervisión) y formas de adquisición de datos con comunicación serial[5].

Figura 10. Ejemplos de software de supervisión.



Fuente: Olivares Ruiz Gonzalo. Configuración de redes de Telemedida.

Además se investigó sobre los adelantos que trae esta plataforma en esta versión, para ser aprovechados, tales como la orientada a objetos, que permite ligar estos últimos mediante mensajes, para la solución del problema. Es una extensión de la programación estructurada que permite potenciar los conceptos de modularidad y reutilización de código. En la figura 9, se muestra la imagen que muestra esta versión.

2.3 REVISIÓN DOCUMENTAL PARA ESTUDIO Y ANALISIS

De la información Recopilada se pudo constatar cuales fueron los elementos a favor y en contra de nuestro sistema de telemetría, así como se analizó también las experiencias que otros han tenido en el

[5] www.mathworks.com

desarrollo de proyectos de similares concepciones. Con todo lo deducido se encamino nuestro trabajo de grado.

2.4 DISEÑO, IMPLEMENTACIÓN Y PRUEBAS AL MÓDULO DE ADQUISICIÓN DE DATOS

Inicialmente se investigó y se implementó un sistema de adquisición de datos el cual tenía que cumplir con las siguientes características.

- Seis entradas análogas: tres para tensión y tres para corriente (cada una de las fases del generador síncrono).
- Las entradas análogas tenían que ser de 0-5V puesto que es el nivel de tensión TTL, que exige el microcontrolador.

2.5 DISEÑO DEL PROTOCOLO DE COMUNICACIONES MICROCONTOLADOR- COMPUTADOR

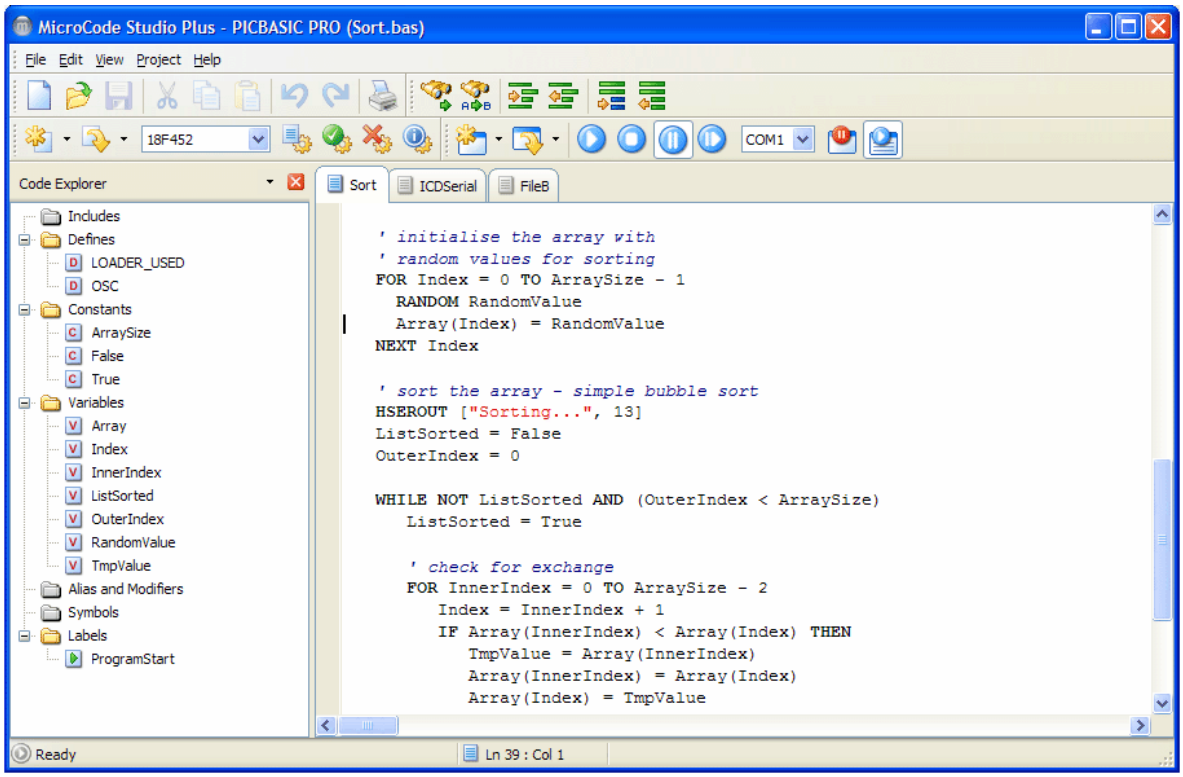
Para el diseño se tuvieron que seleccionar los bits que iban a cumplir la función de datos, paridad, control de flujo, chequeo de error, bits de parada, velocidad de transmisión y trama.

La selección se realizó considerando los parámetros disponibles de configuración de la comunicación serial tanto en Matlab y el Microcontrolador como en el adaptador Bluetooth escogido.

2.6 PROGRAMACIÓN DEL MICROCONTROLADOR

Para programar el microcontrolador se tuvo especial cuidado en el programa a utilizar, se investigó cual podría ser el mejor de los innumerables que existen y se llego a la conclusión que era el PIC Basic Pro Demo, puesto que cumple las características de ser en lenguaje Basic, que es muy fácil de programar y no necesita el lenguaje ensamblador, además esta versión es gratis en Internet[6].

Figura 11. Software Pic Basic Pro Utilizando el editor de texto Microcode Studio Plus.



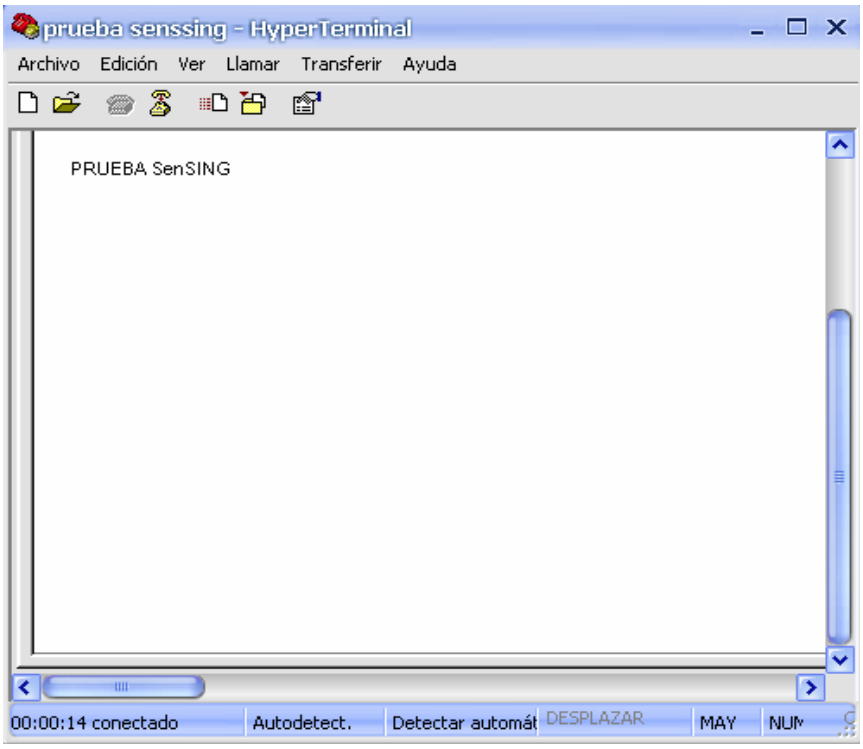
Fuente: http://www.melabs.com/resources/win_ide.htm

[6] http://www.melabs.com/pbpdemo.htm

Se tuvo también que implementar el Microcode Studio que es el editor de texto que utiliza el PIC Basic Pro, como se muestra en la figura 11. También se tuvo que consultar los Datasheets: del microcontrolador 16F877/874 de Microchip (Anexo A).

A partir de las hojas de datos y de ejemplos descargados de Internet[7] se obtuvo la información necesaria sobre la configuración y manejo de los componentes internos del procesador. Inicialmente se realizaron programas de prueba de la conversión A/D y de la comunicación serial. Luego se desarrolló y probó un programa de conversión A/D para seis canales. Finalmente se implementó el programa de comunicación.

Figura 12. Pruebas al microcontrolador utilizando Hyperterminal



Fuente: Microsoft Windows XP

[7] www.todopic.com.arg

El envío de datos desde el microcontrolador, así como la conversión A/D, se verificó en el computador mediante Hyperterminal, como se muestra en la figura 12. De esta manera se comprobó la velocidad de transmisión, el formato de trama, la frecuencia de muestreo y los resultados de la conversión. (Anexo G).

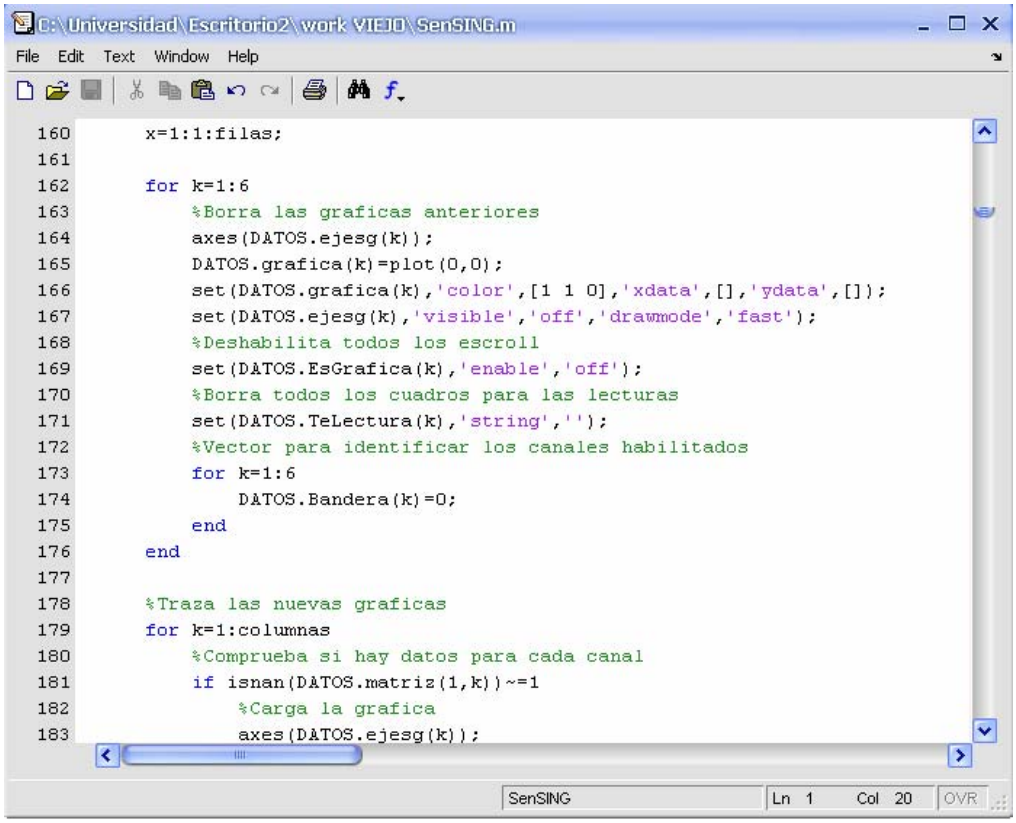
2.7 PROGRAMACIÓN EN MATLAB 7.0.4

Para el diseño de la interfaz gráfica y el programa de comunicación se siguieron los pasos que se indican a continuación (Anexo H):

- Diseño de cada uno de las pantallas que conforman el programa.
- Selección de un programa de diseño gráfico y desarrollo de las imágenes requeridas por la aplicación.
- Programación de las funciones asociadas a los diferentes controles de la interfaz gráfica.
- Pruebas preliminares de transmisión y recepción de datos con Matlab.
- Integración de todas las funciones, vinculación de pantallas y pruebas.

En la figura 13 se muestra el editor de programación de MATALB 7.0.4 con algunas de las líneas del código fuente desarrollado para el sistema de telemetría.

Figura 13. Editor de programación de MATLAB 7.0.4



Fuente: MATLAB 7.0.4

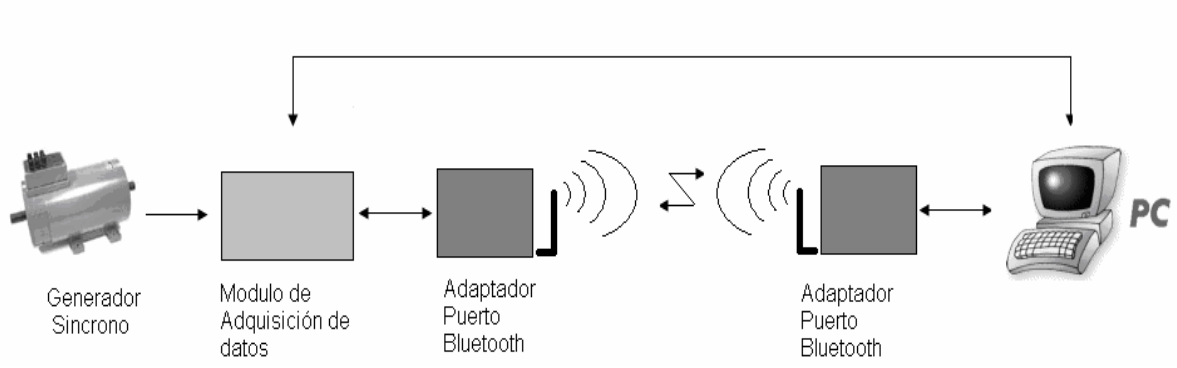
2.8 PRUEBAS DEL SISTEMA

Integración del sistema y pruebas finales. Para esto se realizó el enlace entre el módulo de adquisición de datos y el computador y luego se tomaron algunas señales de entrada para ser representadas gráficamente en Matlab.

3 RESULTADOS Y ANÁLISIS

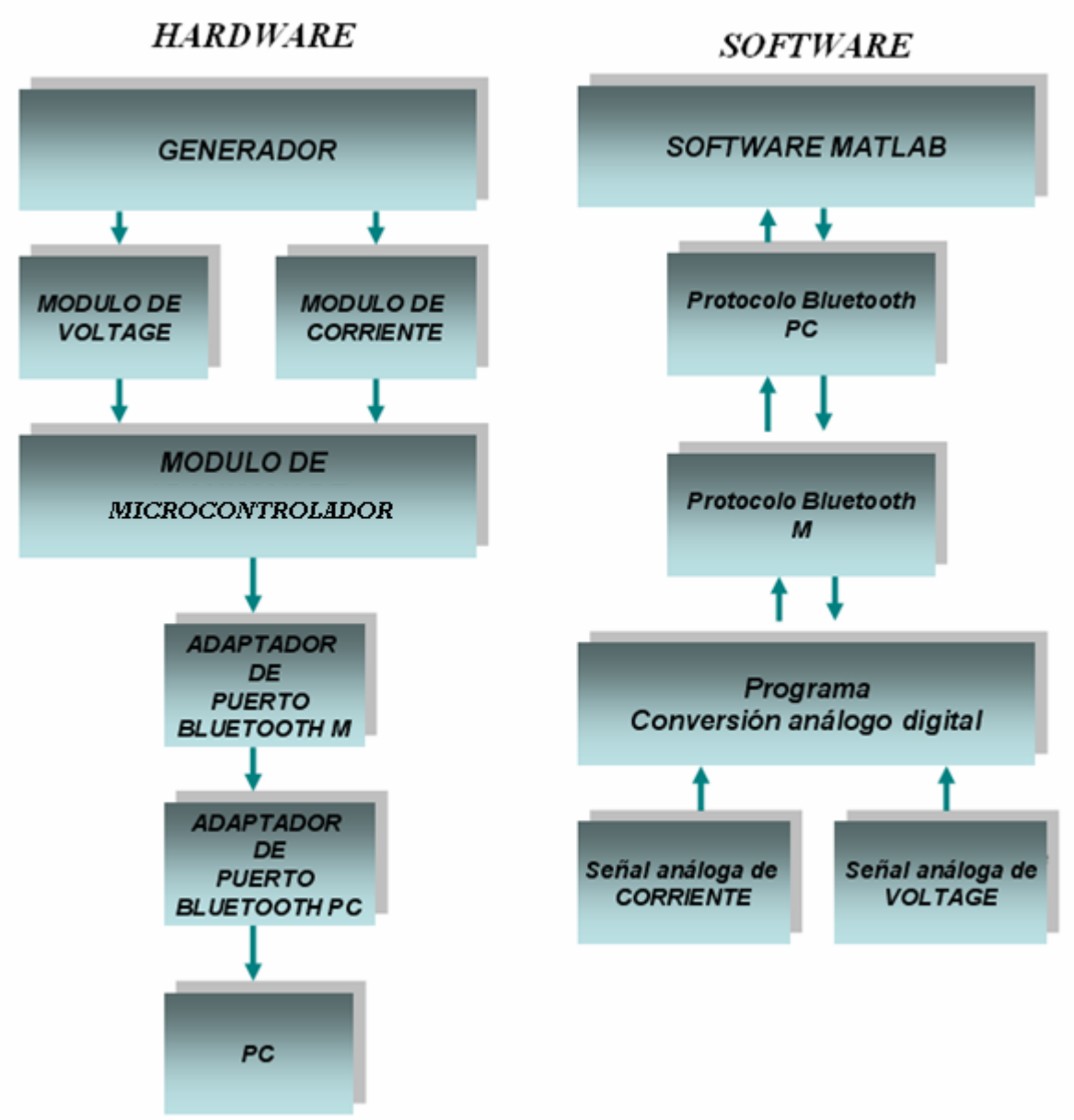
El sistema de monitoreo tiene dos elementos principales: un módulo de adquisición de datos dotado de seis entradas análogas, tres de corriente y tres de voltaje, y un programa para la configuración del sistema y la visualización de los datos. Se requiere un enlace entre estos dos componentes, el enlace inalámbrico requiere dos adaptadores de puerto Bluetooth o un enlace alternativo utilizando un cable serial (Figura 14).

Figura 14. Diagrama funcional del Sistema de monitoreo.



Para mayor entendimiento del sistema de telemetría se presenta un diagrama de bloques de hardware y un diagrama de bloques del software (Figura 15).

Figura 15. Diagrama de bloques del Hardware y Software

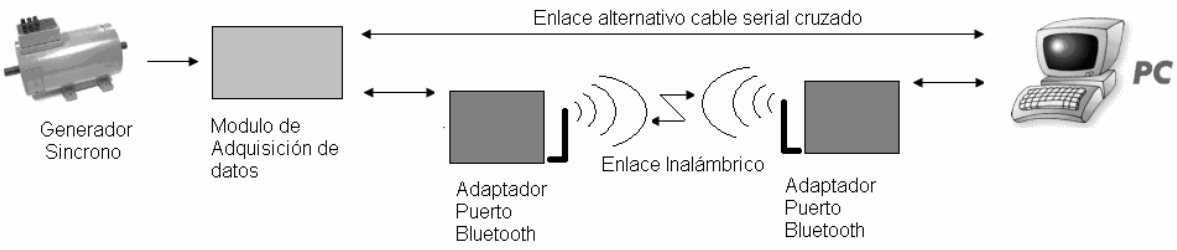


A continuación se realiza una descripción de cada uno de los elementos del sistema.

3.1 COMUNICACIONES

Para las comunicaciones entre el módulo de adquisición de datos y el adaptador de puerto Bluetooth, y entre el adaptador y el computador (Figura 16), se utilizó la interfaz EIA/TIA 232D, conocida normalmente como RS-232.

Figura 16. Enlaces de datos.



El uso de esta interfaz da flexibilidad a la aplicación ya que permite que otros dispositivos, diferentes al módulo de adquisición de datos, sean usados con el adaptador de puerto; además permite reemplazar el enlace inalámbrico por un cable serial.

A continuación se realiza una descripción detallada de la comunicación del sistema.

3.1.1 Carácter

La comunicación es serial y asíncrona. Para indicar el comienzo de cada carácter se envía un bit de arranque. Luego se envían 8 bits de datos y finalmente se transmite un bit de parada para regresar la línea de datos a su estado "ocioso". No se envía bit de paridad (Figura 17).

Figura 17. Formato de carácter.

3.1.2 Trama

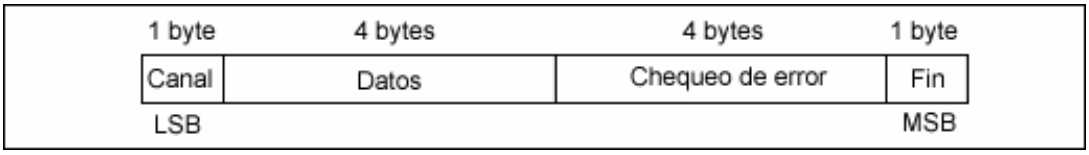
La longitud de trama es variable y depende del tipo de mensaje. En general, se tienen tres tipos de mensajes: configuración, comando y datos. Los mensajes de configuración son intercambiados entre el computador y los adaptadores de puerto Bluetooth. Los comandos son transmitidos desde el computador al Módulo de Adquisición de Datos (Tabla 4).

Tabla 4.Comandos transmitidos desde el computador para el Módulo de Adquisición de Datos.

Mensajes	Formato
Habilitar canal #	<#>
Habilitar todos los canales	<7>
Iniciar muestreo	<I>
Detener muestreo	<D>

La trama de datos, correspondiente al valor de una de las entradas análogas, tiene una longitud de 10 caracteres y está conformada como se indica en la Figura 18.

Figura 18. Formato de la trama de datos.

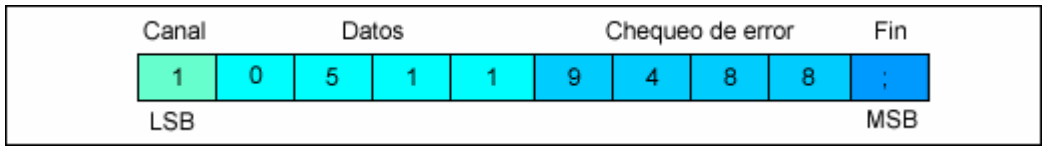


- El canal, un número entre 1 y 6, corresponde a la entrada del módulo a la que se conectó la señal.
- El dato, un número entre 0 y 1023, corresponde al valor obtenido por el microcontrolador en la conversión A/D del canal correspondiente. Este número está codificado de tal manera que el byte más significativo (MSB) del campo de datos corresponde a las unidades de mil y los siguientes bytes, de izquierda a derecha, son las centenas, decenas y unidades, respectivamente.
- El campo de chequeo de error contiene la diferencia entre el campo de datos y 9999. Los bytes en este campo se organizan de manera similar que los del campo de datos.
- El carácter de fin de trama es ";".

Por ejemplo, si en el canal 1 hay un voltaje de entrada de 2.5 V entonces se tendrá 511 como resultado de la conversión A/D. El campo de datos estará formado por los caracteres "0", "5", "1" y "1", el campo de chequeo de error estará formado por los caracteres resultantes de

restar 511 de 9999; es decir, "9", "4", "8" y "8". La trama resultante se presenta en la Figura 19.

Figura 19. Ejemplo de trama de datos.



3.1.3 Control de flujo.

La comunicación serial es semidúplex, con las siguientes particularidades:

- Los comandos transmitidos desde el computador son confirmados por el módulo de adquisición de datos. En este caso, la retransmisión es implícita; es decir, el computador transmite nuevamente el comando cuando no recibe confirmación del módulo.
- Los datos enviados desde el módulo no son confirmados y nunca se retransmiten.

El control de flujo es Parada-Espera, de esta forma se evitan posibles colisiones en el canal de comunicación.

3.1.4 Velocidad de transmisión.

La velocidad de transmisión es configurable. La velocidad por defecto es de 19200 bps; sin embargo, pueden utilizarse velocidades desde 1200 bps.

A continuación se presenta el cálculo de la velocidad:

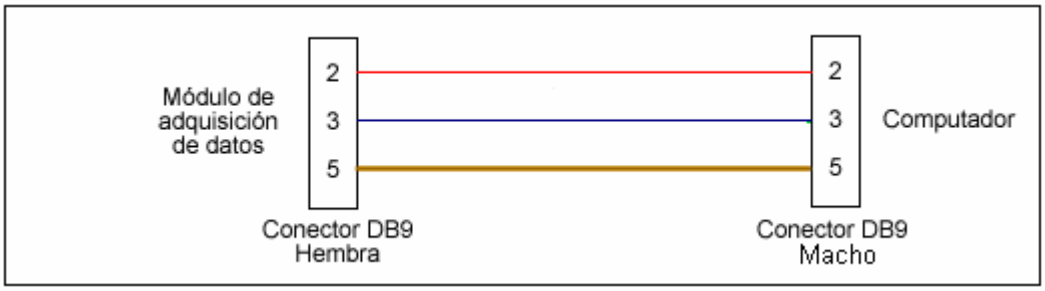
$$Velocidad \quad [bps] = Numero \ de \ canales * \frac{20 \ caracteres}{canal * s} * \frac{10 \ bits}{caracter}$$

- Número de canales: 1 a 6 canales, configurables desde el computador.
- Número de caracteres: 10 caracteres por trama o por lectura. En cada canal se realizan dos lecturas por segundo.
- Número de bits: la trama serial utilizada usa 10 bits por carácter.

3.1.5 Cable serial.

El enlace inalámbrico puede ser reemplazado por un cable serial. La asignación de pines sigue el estándar EIA/TIA 232D (Figura 20).

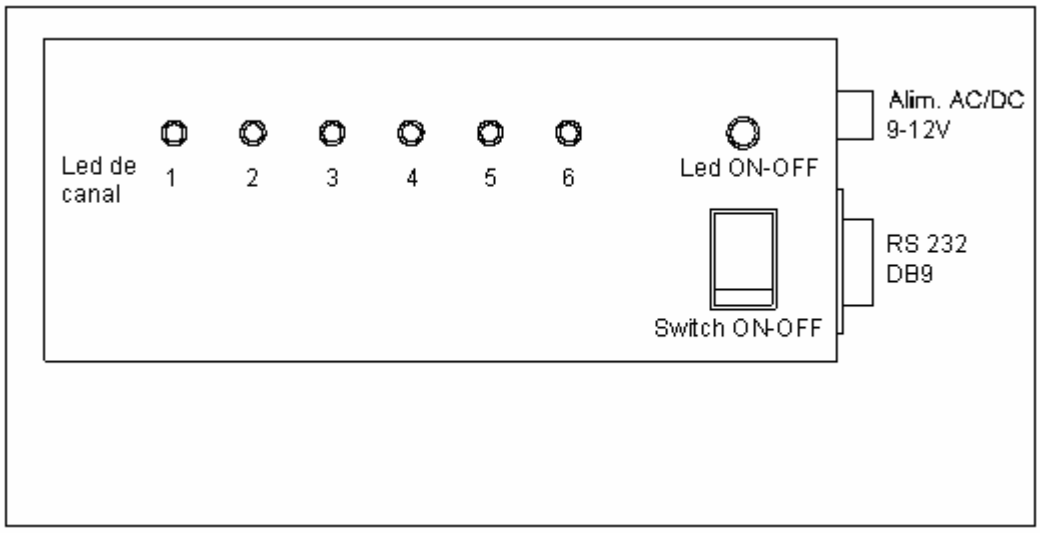
Figura 20. Configuración de un cable serial para el enlace Módulo de adquisición-Computador.



3.2 MÓDULO DE ADQUISICIÓN DE DATOS

3.2.1 Hardware.

Figura 21. Modulo de adquisición de datos



El módulo de adquisición de datos (Figura 21) cuenta con las siguientes características:

- Una entrada de alimentación de a 12-20 V la cual debe ser DC, con el fin de poder utilizar un adaptador de voltaje comercial.
- Un puerto serie RS 232 para la comunicación con el equipo visualizador de datos (PC), para la conexión directa entre estos se debe utilizar un cable serial.
- Un suitche para encendido del módulo
- Un led indicador de encendido.
- 4 entradas análogas de voltaje 0-115 V, 3 para cada una de las fases y una para el neutro (canales 1 a 3). Y 6 para corriente,

canales 4 a 6, pueden configurarse como entradas de corriente de 0-1.4 A.

- Leds indicadores de los canales habilitados.

El módulo cuenta con un seguidor de voltaje en cada uno de los canales (Figura 22 y 23) con el fin de brindar entradas de alta impedancia y no afectar la medición de manera significativa.

Figura 22. Entrada de alta impedancia.

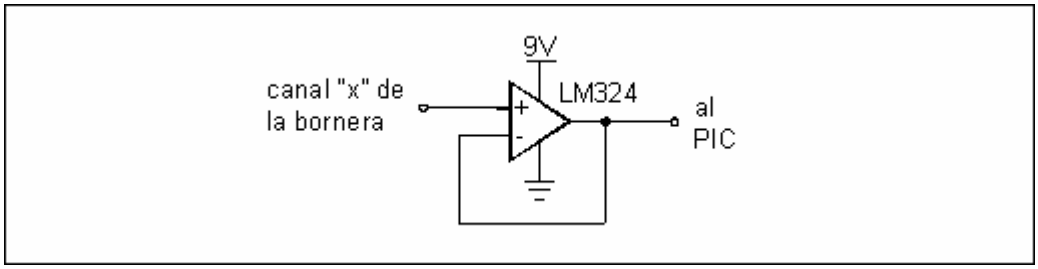
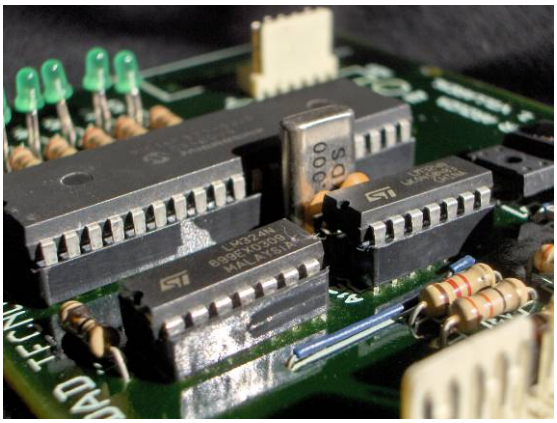


Figura 23. Vista de los AMP-OP



Para establecer la comunicación Módulo-Computador mediante la interfaz RS-232 es necesario adecuar los niveles de voltaje TTL manejados por el procesador a los manejados en la transmisión serial. Esto se realizó mediante un driver MAX232 (Figura 24 Y 25).

Figura 24. Driver MAX232.

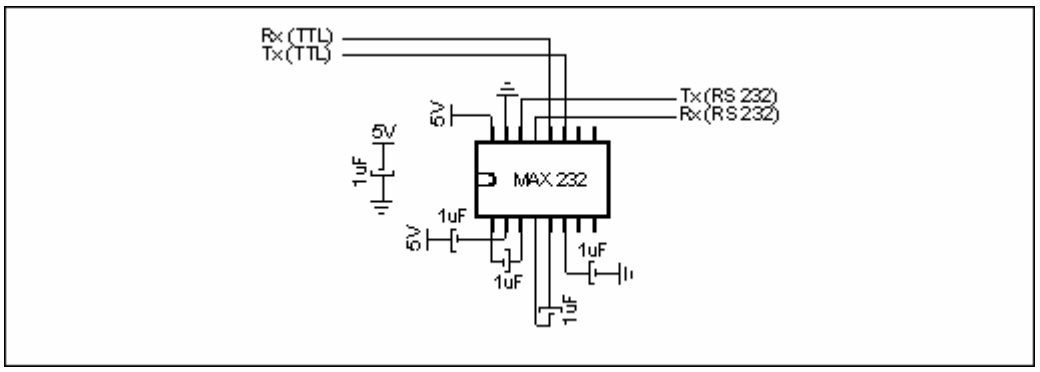
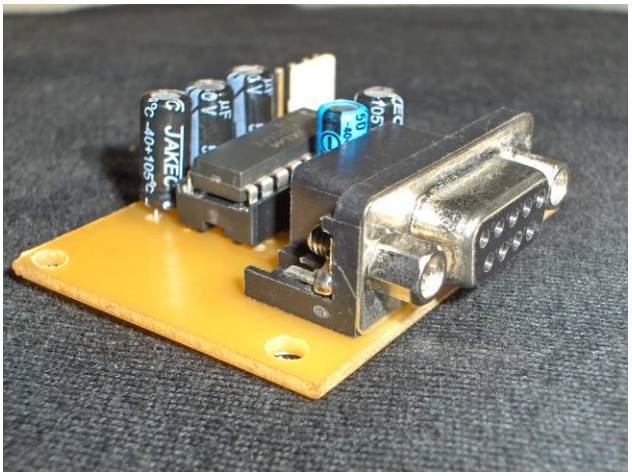


Figura 25. Vista de MAX232



El procesador utilizado en el módulo es el PIC 16F877-20, como se muestra en la figura 26 de la gama media de los microcontroladores de la marca MICROCHIP (Anexo A). Se escogió basado en los siguientes parámetros requeridos por el diseño:

- Entradas A/D, cuenta con 8 entradas análogas con resolución de 10 bits en la conversión.
- Alta velocidad de operación, trabaja a 20Mhz, la más alta en los procesadores de la serie 16Fxxx de MICROCHIP.

- Número de entradas y salidas suficientes para el manejo de la electrónica asociada. Posee 33 pines configurables como entrada/salida los cuales son suficientes para la aplicación.
- Cuenta con una USART para el manejo de las comunicaciones seriales y además posee dos características importantes para la programación que son: buffer de 2 bytes en la recepción e interrupción por llegada de carácter al buffer.

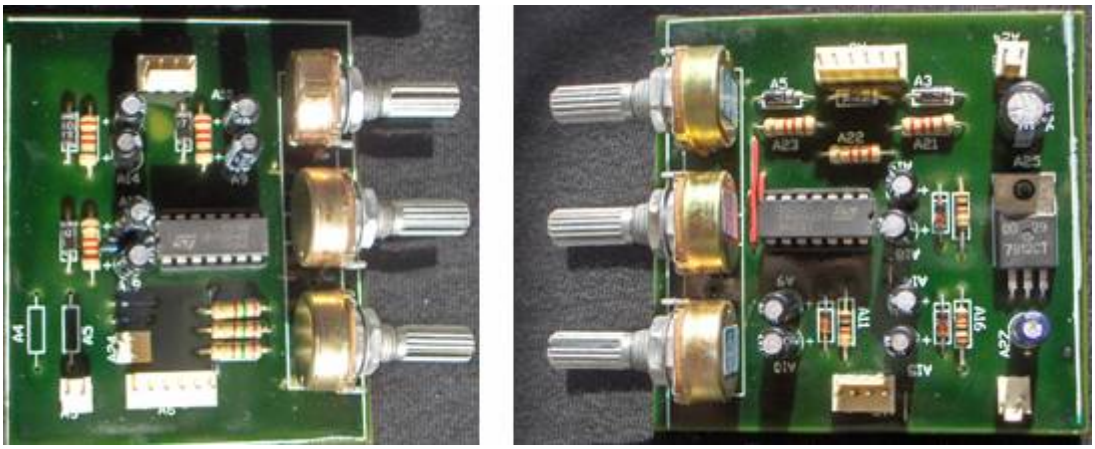
Figura 26. Modulo de Microcontrolador

Para los tres canales de medición de corriente se utilizan dos resistencias comerciales de 220Ω y 33Ω en serie, con el fin de cerrar el loop y generar una señal de voltaje proporcional a dicha corriente de entrada, la cual es muestreada por el PIC.

Para monitorear los voltajes de línea y las corrientes de carga del generador sincrónico (ver anexo F), se implementaron dos circuitos independientes para el acondicionamiento de las señales (ver anexos D y E).

La calibración de los módulos se realiza mediante potenciómetros de 5KΩ del módulo de corriente (ver anexo D) y de 20KΩ del módulo de voltaje (ver anexo E). Estos permiten ajustar el valor entregado de corriente y voltaje al valor RMS medido con un multímetro digital. En la figura 27 se muestran los módulos de corriente y voltaje respectivamente.

Figura 27. Modulo de Corriente y voltaje respectivamente



Los tres transformadores de potencial implementados tienen las siguientes características:

- **Relación de transformación**: 115/12 VAC
- **Corriente**: 200mA
- **Frecuencia**: 60 Hz

Los tres transformadores de corriente implementados tienen las siguientes características:

- **Relación de transformación**: 50/5 A
- **VA**: 1.41
- **Frecuencia**: 60 Hz

3.2.2 Software

3.2.2.1 Entradas/salidas:

En el procesador se configuraron todos los pines de los puertos A y E como entradas (Figura 28) ya que allí es donde se encuentra el hardware de conversión A/D del procesador; el resto de pines se configuran como salidas:

Figura 28. Configuración de los puertos de E/S

```
TRISA = %00011111
TRISB = %00000000
TRISC = %10000000
TRISD = %00000000
TRISE = %00000111
```

Los pines del procesador configurados como entradas y que no son utilizados se llevan al voltaje de referencia (0 V) para evitar que el ruido en éstos afecte el funcionamiento del microcontrolador.

3.2.2.2 Conversor A/D:

Para la utilización del conversor A/D se deben configurar dos registros del procesador (Figura 29). Un registro controla la velocidad de conversión, el canal a muestrear, la activación del módulo de conversión y la inicialización de dicho módulo; el otro registro controla la justificación del resultado de la conversión, también se activan los pines del procesador que se utilizan como entradas análogas y como voltajes de referencia: se configuraron todos los pines como entradas y los voltajes de referencia son los de alimentación del procesador.

Figura 29. Configuración de registros para la conversión A/D.

```
ADCON0 = %10000001
ADCON1 = %10000000
```

3.2.2.3 Usart.

Para la utilización de la USART se deben configurar los siguientes registros:

- RCSTA: en este registro se activa o desactiva la utilización de la USART, se activa la recepción continua de caracteres, se activa o desactiva la recepción de un noveno bit (que puede ser utilizado como paridad), se define el formato. Se habilitó la USART en recepción continua y formato 8-N-1.

- TXSTA: en este registro se controla la fuente de reloj (interna o externa), la transmisión del noveno bit, se activa la transmisión en la USART, se selecciona el modo de funcionamiento (síncrono o asíncrono), y se controla la rata de baudios en la USART (alta o

baja).Se seleccionó transmisión continua en formato 8-N-1 y con rata de baudios alta.

- SPBRG: en este registro se coloca un valor hallado en tablas el cual define la velocidad de la transmisión Se escogió "64" por que con una frecuencia de 20Mhz, rata de baudios alta y SPBRG = 64, se tiene una tasa de baudios de 19200 bits/s la cual es lo suficientemente alta como para enviar los datos serialmente sin demora significativa.

En la figura 30 se presenta la configuración de los registros de la USART para el Sistema de monitoreo.

Figura 30. Registros de configuración de la USART

```
RCSTA = %10010000
TXSTA = %00100100
SPBRG = 64
```

3.2.2.4 Interrupciones.

La recepción de datos en el PIC se realiza mediante interrupciones. Para configurarlas es necesario habilitar las interrupciones de periféricos y por llegada de caracteres (Figura 31) como se indica a continuación:

- PIE1.5: con la activación de este bit se habilita la interrupción por llegada de caracter a la USART.

- INTCON: en este registro se habilita la interrupción de periféricos, para que la USART pueda generar la interrupción de caracter.

Figura 31. Configuración de la interrupción por recepción de carácter

```
PIE1.5 = 1
INTCON = %11000000
```

3.2.2.5 Registros complementarios

Además de los registros descritos, se deben utilizar ciertas definiciones, las cuales terminan de configurar el microcontrolador para la adquisición de los datos. Los registros adicionales se describen a continuación:

- DEFINE OSC 20: con esta definición se configura la velocidad del oscilador con el que trabajará el procesador.
- DEFINE ADC_BITS 10: con esta definición se configura la resolución de la conversión A/D.
- DEFINE ADC_SAMPLEUS 50: con esta definición se selecciona un tiempo de muestreo de 50µs.

3.2.3 Funcionamiento

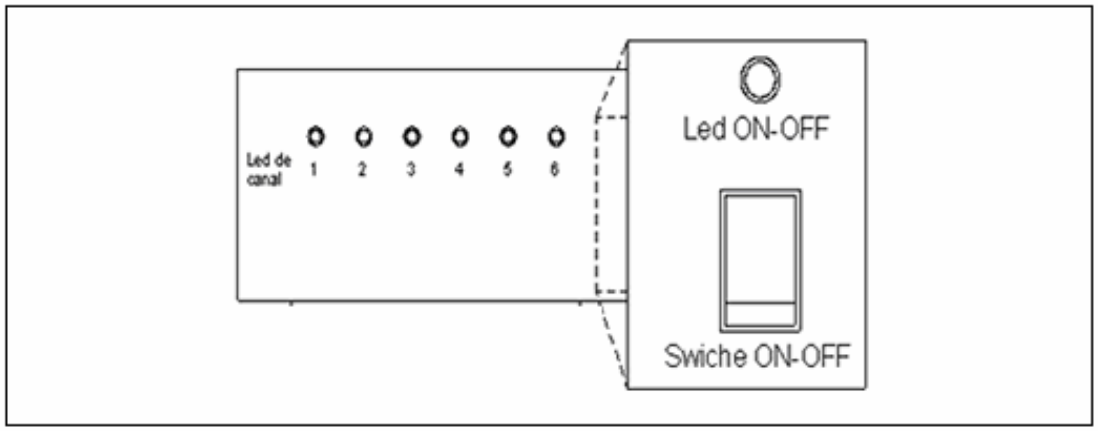
Para poder utilizar el módulo de adquisición de datos deben tenerse en cuenta las siguientes condiciones:

- Todas las conexiones del cableado deben hacerse con el módulo apagado con el fin de evitar accidentes que puedan causar el deterioro del módulo, del equipo o circuito que se pretenda conectar al mismo.
- Se debe tener en cuenta el rango de voltajes en la alimentación del modulo, con el fin de lograr un correcto funcionamiento del mismo y no correr el riesgo de dañarlo.

- Si no se utiliza el enlace serial inalámbrico, la alternativa es un cable serial.
- El módulo cuenta con una bornera de 10 entradas. 4 entradas análogas de voltaje 0-115 V, 3 para cada una de las fases y una para el neutro (canales 1 a 3). Y 6 para corriente, canales 4 a 6, pueden configurarse como entradas de corriente de 0-1.4 A.

Después de cablear las señales de entrada al módulo y de establecer el enlace serial, se debe encender el módulo. Para esto está provisto de un suiche ON/OFF y de un led indicador (figura 32).

Figura 32. Encendido del modulo

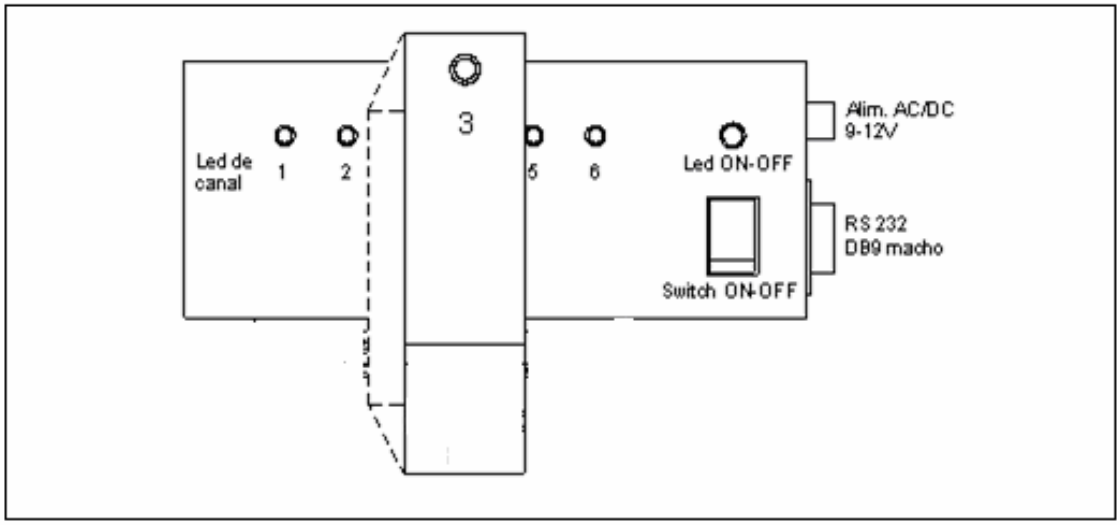


A partir de este momento el módulo está a la espera de comandos de identificación, configuración e inicio del muestreo de las señales de entrada, emitidas a través de la interfaz gráfica de Matlab.

El primer comando que recibe el módulo es el de identificación, luego de responder a éste entra al modo de configuración en el que es posible habilitar canales de entrada.

La habilitación de un canal en particular es señalada por el led correspondiente (figura 33).

Figura 33. Led indicador de canal habilitado/uso



Cuando el procesador recibe el comando de inicio, comienza a realizar la conversión A/D en el(los) canal(es) que ha(n) sido activado(s) previamente; el procesador envía el resultado de esta conversión a través de la USART dos veces por segundo (independientemente del número de canales activos). Esto lo hace hasta que el programa de Matlab envíe el comando de parada.

3.3 ADAPTADOR DE PUERTO BLUETOOTH

3.3.1 Características técnicas

A continuación se describen algunas características del adaptador de puerto seleccionado (Anexo B) como se muestra en la figura 34.

Figura 34. Adaptador de puerto Bluetooth

Fuente: www.bradatech.com

3.3.1.1 Velocidad de transmisión

El adaptador de puerto dispone de velocidades desde 1200bps hasta 115200bps. La velocidad por defecto para el sistema de monitoreo es de 19600 bps. Esta velocidad debe configurarse tanto en el adaptador de puerto como en el PIC y el computador.

3.3.1.2 Control de flujo

El estado de los pines RTS y CTS de la interfaz RS-232 es intercambiado inalámbricamente, lo que permite realizar control de flujo por hardware. Este control de flujo pretende evitar desbordamiento y pérdida de datos en el buffer de recepción.

3.3.1.3 Bajo consumo de potencia

Esta es una de las características más importantes de los dispositivos Bluetooth. Los chips son pequeños y de bajo consumo de potencia para ser encajados en cualquier equipo o ser usados en dispositivos portátiles.

El adaptador de puerto Bluetooth requiere un voltaje de alimentación de 9 VDC.

3.3.1.4 Conectores

El adaptador está provisto de un conector DB9 hembra, un adaptador "*No Hardware Handshaking* adapter" para la conexión al computador y para nuestro modulo un cable serial. Esta característica le permite ser conectado al puerto serial estándar de cualquier dispositivo.

3.3.2 Configuración

Para hacer la configuración el fabricante dispone de un cd que contiene el software y su manual de programación.

Los módulos se encuentran programados siguiendo las instrucciones del fabricante, esto se hace solo una vez, con lo cual los adaptadores bluetooth se mantienen programados aun cuando no tengan las baterías. Por lo anterior los adaptadores son identificados con la iniciales PC para el adaptador a conectar en el Computador y la letra M para el adaptador bluetooth a conectar al modulo de adquisición de datos.

3.4 INTERFAZ GRÁFICA DE USUARIO

3.4.1 Descripción

La interfaz gráfica en Matlab está diseñada para facilitar la interacción con el módulo de adquisición de datos a través del enlace inalámbrico propuesto o de un cable serial; además de permitir la visualización y registro de hasta seis señales análogas.

La interfaz se implementó en Matlab para permitir la integración del sistema de monitoreo con las múltiples y poderosas herramientas de análisis y simulación que presenta este programa.

El código desarrollado presenta las siguientes características:

- Orientado a objetos: la programación utiliza objetos, ligados mediante mensajes, para la solución del problema. Es una extensión de la programación estructurada que permite potenciar los conceptos de modularidad y reutilización de código.
- Objetos: los objetos están representados por los controles de la interfaz gráfica y por los controles internos como la interfaz software y el objeto serial. Estos objetos poseen propiedades y eventos definidos, cuya interacción definen el programa.
- Mensajes: cuando se ejecuta el programa, los objetos están interpretando y respondiendo a mensajes.
- Modularidad: el programa se ha dividido en funciones, pueden agregarse, moverse y quitarse funciones sin afectar el código restante.
- Reutilización: el código define la manera como se modifica las propiedades de los objetos a partir de mensajes o eventos, esto hace que pueda tomarse una función definida para un control e insertarse sin problemas en un nuevo código.

A continuación se describe cada uno de los elementos que conforman la interfaz gráfica de usuario.

Menú Archivo

- Abrir: permite abrir un archivo existente y visualizar las gráficas correspondientes. Los archivos válidos tienen un formato particular y pueden crearse guardando las secciones de monitoreo o tomando datos en el formato generado por la aplicación. Todos estos archivos son en Excel.
- Nuevo: reinicializa toda la aplicación, incluyendo la configuración del módulo de adquisición de datos.
- Salir: cierra el programa.

Menú Opciones

- Canales: permite configurar cada canal del módulo de adquisición de datos. Por cada canal se despliega una ventana modal en la que puede especificarse un nombre para la señal que se monitoreará, los valores máximos y mínimos que toma la señal, los valores máximos y mínimos para la visualización y la habilitación del canal.

  Este submenú está dividido en dos partes. Entradas de corriente y entradas de voltaje. Por cada tipo de entrada se tienen hasta tres canales.
- Iniciar: inicia el monitoreo de las señales conectadas a los canales habilitados.
- Detener: detiene el monitoreo.

Menú Herramientas

- Configurar puerto: permite ajustar el puerto serial y las características de la comunicación como velocidad, bits de datos, paridad y control de flujo.

Menú Ayuda

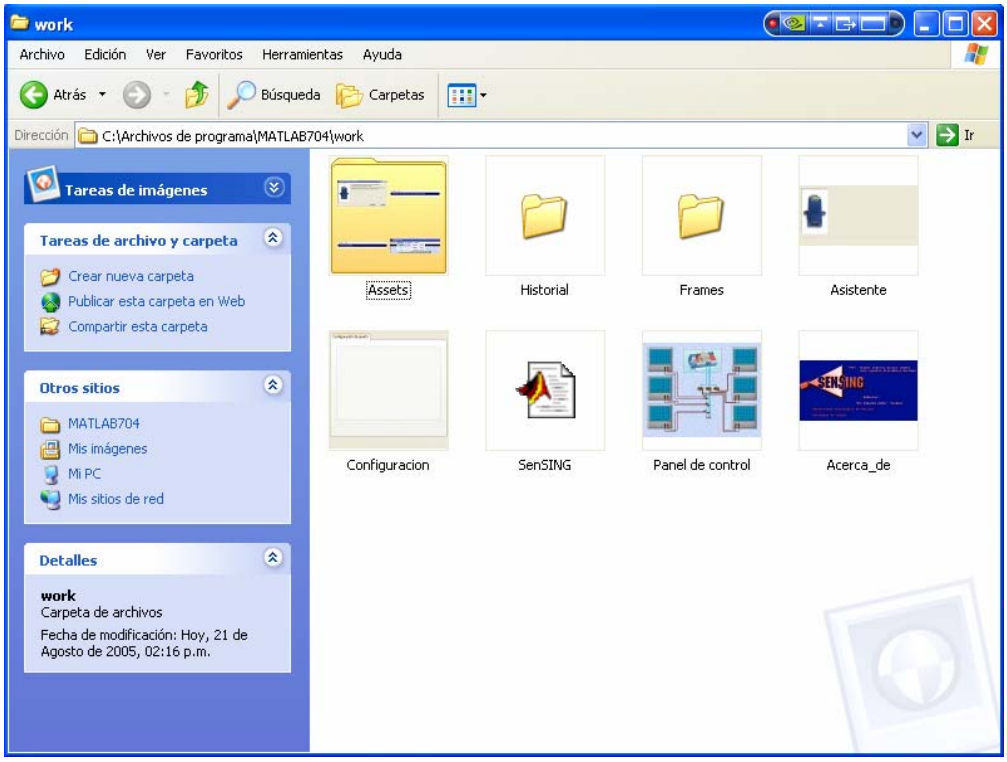
- Ayuda: contiene indicaciones sobre el manejo del programa.
- Acerca de: contiene el nombre del Trabajo Dirigido de Grado, los autores, el director, la versión del programa y la fecha de elaboración.

3.4.2 Instalación

El programa requiere MATLAB, versión 7.0.4 o posterior, un puerto serial disponible y Microsoft Excel.

Primero debe copiar del cd **SenSING** todo lo que se encuentra dentro de la carpeta PROGRAMAS_MATLAB y pegar en la carpeta llamada work de MATLAB. Como se muestra en la figura 35.

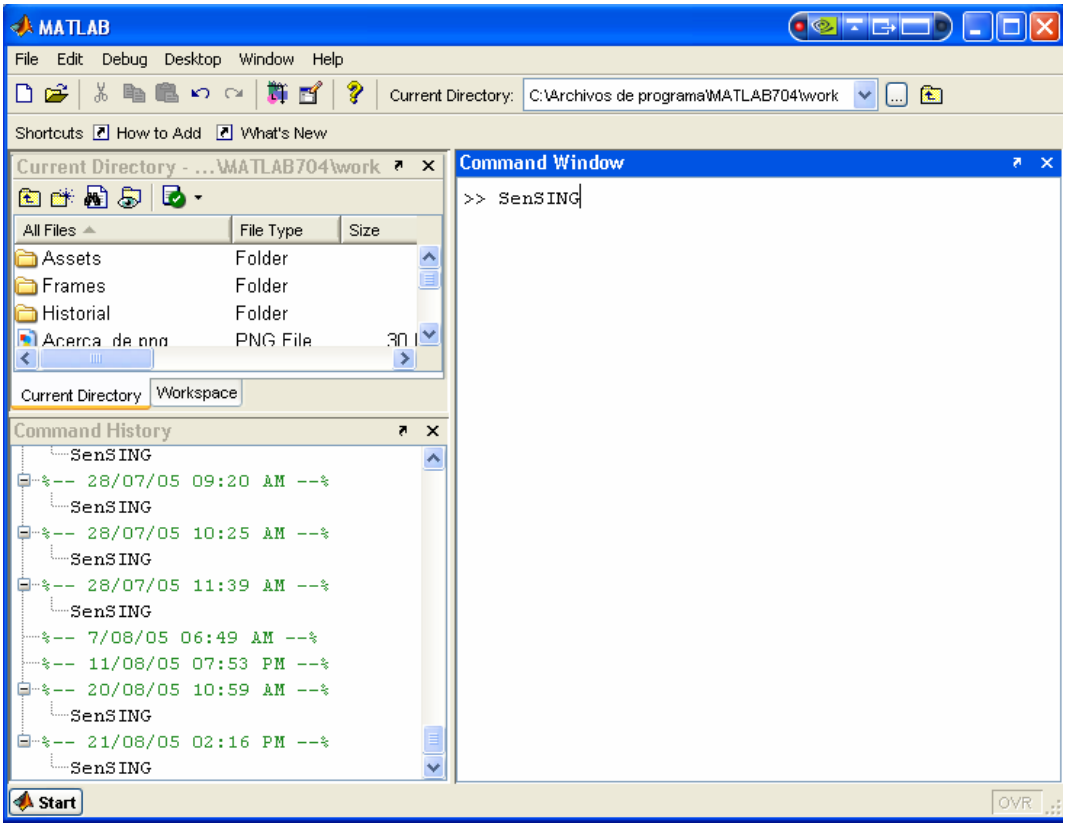
Figura 35. Carpeta work de MATLAB donde se ubican las carpetas.



Después de realizado lo anterior, todo quedara listo para que el programa sea llamado desde MATLAB.

Ahora después de haber abierto MATLAB, se escribe SenSING en el command window y se pulsa la tecla enter, como se muestra en la figura 36.

Figura 36. Llamado del programa desde MATLAB.

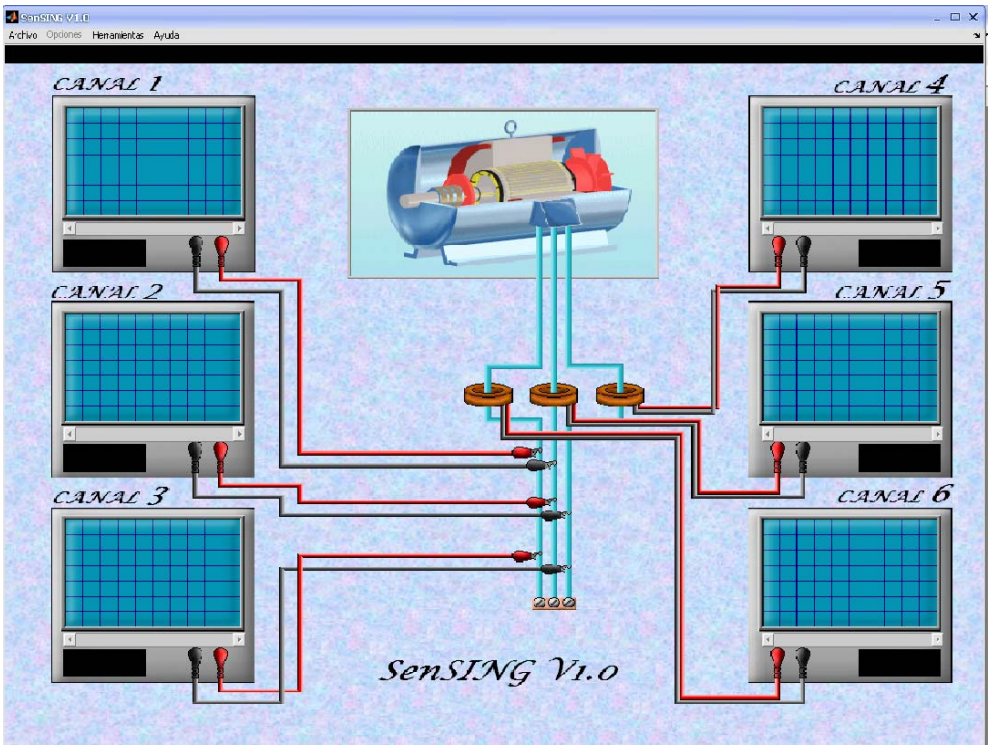


Hecho lo anterior aparecerá la pantalla mostrada en la figura 37.

Para salir de la pantalla "Acerca de" basta con pulsar cualquier tecla.

En este instante se tiene deshabilitada toda la interfaz, incluyendo el comando para iniciar muestreo en el menú Opciones.

Figura 37. Interfaz grafica de usuario



3.4.3 Operación

Los pasos que deben seguirse para iniciar una sección de monitoreo son:

- Encender el módulo de adquisición de datos.
- Realizar el enlace, bien sea conectando un cable serial entre el computador y el módulo de adquisición de datos o conectando el par de adaptadores de puerto Bluetooth. Se recomiendan distancias máximas de 30m para el adaptador de puerto, en espacio libre.
- Iniciar el programa desde Matlab.

- Configurar el puerto de comunicaciones. Primero debe seleccionarse un puerto libre del computador, al que se hizo la conexión del cable o del adaptador Bluetooth. Este puerto no debe estar siendo usado por otra aplicación tal como Hyperteminal. Si se utiliza el adaptador de puerto Bluetooth puede configurarse con ayuda del asistente en el menú Herramientas.
- Configurar y habilitar los canales que se utilizarán. Puede usarse cualquier número de canales entre 1 y 6. Los canales de corriente se configuran por separado de los de voltaje y solo pueden utilizarse canales consecutivos de cada tipo.
- Iniciar el muestreo.
- Detener el muestreo.

Todos estos pasos se encuentran de manera mas detallada en la ayuda que se encuentra en la barra de menú del programa.

3.5 CARACTERISTICAS TECNICAS DE SenSING

- **Magnitud a medir.** Corriente y tensión de cada una de las tres fases de un generador sincrono.
- **Rango o margen de la medida.** De 0-115 V para tensión y de 0-1.4 A para la medida de corriente.
- **Precisión de la medida.** Después de comparar las medidas con voltímetros y amperímetros de la marca FLUKE y marca Metex. Para tensión hay un error de más ó menos 1 Voltio y para corriente de más ó menos 0.1 Amperio.
- **Resolución.** (mínimo valor que el medidor puede discriminar). Para la medida de tensión, es de 20V y para la medida de corriente, es de 0.2 A.

- **Salida Digital.** Se realiza mediante los adaptadores de puerto Bluetooth o mediante cable serial.
- **Tensión de Alimentación.** Adaptador de 12-20 V.

4 CONCLUSIONES

La tecnología escogida para realizar la labor de SENSING, es ideal para aplicaciones de monitoreo y control a nivel industrial simplificando la instalación de cableado, reduciendo la probabilidad de falla por cables, empalmes, entre otros y brindando modularidad a los procesos.

Los dispositivos de radio Bluetooth ofrecen excelentes prestaciones para aplicaciones inalámbricas de corto alcance. Entre estas están, alta velocidad de transmisión de datos, seguridad, inmunidad al ruido, protocolo robusto, incluyendo corrección de errores y encriptación.

El módulo de adquisición de datos es un equipo económico y de grandes prestaciones en para el monitoreo de señales ya que está provisto de entradas de voltaje y corriente estándar que pueden muestrear señales de baja frecuencia con una resolución adecuada para la mayoría de los procesos.

El módulo es abierto, es decir, el valor resultante de la conversión A/D para cada uno de los canales, está disponible en un puerto RS232 estándar para ser transmitido a cualquier dispositivo de control, de presentación de datos, de almacenamiento o incluso a través de una red.

El sistema de monitoreo puede usarse en gran variedad de practicas de laboratorio en áreas tan diversas como electrónica, microprocesadores, máquinas, control, etc., donde se capturan señales análogas para su posterior análisis.

El código en Matlab, por estar orientado a objetos, permite la implementación de nuevas funciones para el procesamiento de los datos, sin afectar el código existente.

El sistema es flexible y escalable; es decir, pueden realizarse nuevas conexiones entre diferentes dispositivos, integrar varios módulos y procesos con modificaciones menores del código fuente.

El sistema permite la centralización de la información ya que pueden monitorearse desde el mismo equipo y a través de una sola interfaz las diferentes variables del proceso.

Con todo lo anterior queda ya realizada la parte que corresponde a la supervisión. Uno de los elementos de la tele operación como se mencionó en el prologo. Queda ahora seguir con el de control y unificación.

RECOMENDACIONES

- Dar continuidad al estudio de las áreas de trabajo exploradas en el presente Sistema de telemetría, éstas son: uso de los microcontroladores en aplicaciones de monitoreo y avanzar al control, comunicaciones inalámbricas con nuevas tecnologías que ofrecen enormes ventajas de interoperabilidad e inmunidad al ruido, desarrollo de interfaces gráficas de usuario y comunicaciones con el computador.

- Utilizar el sistema desarrollado en prácticas, con diversos enfoques:
    - Programación: implementar nuevas funciones de visualización, de generación de reportes y de interfaz software a otras herramientas de análisis y presentación de datos.
    - Comunicaciones: analizar y mejorar el protocolo de comunicaciones propuesto y hacerlo extensivo al manejo de redes.
    - Control: extender la funcionalidad del sistema al procesamiento de los datos y control a partir de las variables medidas.
    - Microprocesadores: optimizar el código para obtener frecuencias de muestreo altas, que permitan la captura de eventos o el muestreo de algunos fenómenos transitorios.

BIBLIOGRAFÍA

[1]. Descripción y explicación Online. Octubre de 2004. http://www.zonabluetooth.com/que_es_bluetooth2.htm

[2]. Paper Online. Enero de 2005. http://www.ericsson.com/about/publications/review/1998_03/files/1998031.pdf

[3]. Pagina web. Febrero de 2005.www.bluetooth.org

[4]. Pagina web. Febrero de 2005.www.bluetooth.com

[5]. Pagina web. Marzo de 2005. www.microchips.com

[6]. Programas Online. Marzo de 2005. http://www.melabs.com/pbpdemo.htm

[7]. Manual y ayuda Online. Marzo de 2005.www.todopic.com.arg

[8]. Olivares Ruiz Gonzalo. Configuración de redes de Telemedida.

[9]. BLUETOOTH, Specification of the Bluetooth System version 1.1. Special Group Interesting. Febrero de 2001.

[10]. BALCELLS, Joseph. ROMERAL, José Luis. AUTÓMATAS PROGRAMABLES. Mexico, Alfa omega, 1ra edición.

[11]. COUGHLIN Robert. DRISCOLL Frederick. Amplificadores operacionales y circuitos integrados lineales 5ª edición. Pearson, 1999.

[12]. CEBALLOS, Francisco Javier. Visual Basic Curso de programación. AlfaOmega, México 1998.

[13]. HAARTSEN, Jaap. Bluetooth – El interfaz de radio universal para conectividad ad hoc sin hilos. Ericsson review, volumen 3, 1998.

[14]. ARFWEDSON, Henrik y Rob Sneddon. Módulos Bluetooth de Ericsson. Ericsson review, volumen 4, 1999.

[15]. Página de MovilGate http://www.chilemovil.com/cm/telemetry/faqs.html

[16]. J. Zhang, L. Cheng, and I. Marsic, Models for Non-intrusive Estimation of Wireless Link Bandwidth, En *Proceedings of the*

Personal Wireless Communication Conference (PWC 2003), Venice, Italy, September 23-25, 2003.

[17]. MathWorks. Help: Creating Graphical User Interfaces. MATLAB versión 6.5 R13.

[18]. MathWorks. Help: Programming and Data Types. MATLAB versión 6.5 R13.

[19]. MathWorks. Help: Graphics. MATLAB versión 7.0.4 R14.

[20]. MathWorks. Help: External Interfaces/API. MATLAB versión 7.0.4 R14.

[21]. MathWorks. Help: Data Acquisition Toolbox. MATLAB versión 7.0.4 R14.

[22]. MONTOYA Rafael. CARMONA Roger.Monitoreo inalambrico de corrientes de carga y volatajes de linea de un generador trifasico utilizando Bluetooth. Universidad Nacional de Colombia, 2004.

ANEXO A. Hoja de datos del microcontrolador PIC 16f877-20

MICROCHIP

PIC16F87X

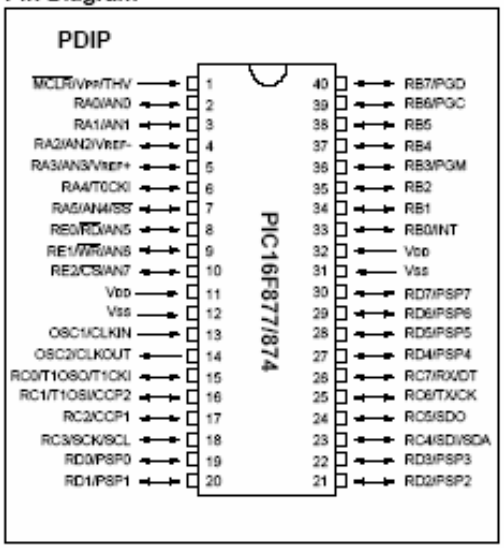
28/40-pin 8-Bit CMOS FLASH Microcontrollers

Microcontroller Core Features:

- High-performance RISC CPU
- Only 35 single word instructions to learn
- All single cycle instructions except for program branches which are two cycle
- Operating speed: DC - 20 MHz clock input
  DC - 200 ns instruction cycle
- Up to 8K x 14 words of FLASH Program Memory,
  Up to 368 x 8 bytes of Data Memory (RAM)
  Up to 256 x 8 bytes of EEPROM data memory
- ★ **Pinout compatible to the PIC16C73/74/76/77**
- Interrupt capability (up to 14 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
- Programmable code-protection
- Power saving SLEEP mode
- Selectable oscillator options
- Low-power, high-speed CMOS FLASH/EEPROM technology
- Fully static design
- In-Circuit Serial Programming™ via two pins
- ★ Only single 5V source needed for programming
- ★ In-Circuit Debugging via two pins
- Processor read/write access to program memory
- Wide operating voltage range: 2.0V to 5.5V
- High Sink/Source Current: 25 mA
- Commercial and Industrial temperature ranges
- Low-power consumption:
  - < 2 mA typical @ 5V, 4 MHz
  - 20 µA typical @ 3V, 32 kHz
  - < 1 µA typical standby current

Pin Diagram

PDIP



Peripheral Features:

- 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
- Two Capture, Compare, PWM modules
- Capture is 16-bit, max. resolution is 12.5 ns, Compare is 16-bit, max. resolution is 200 ns, PWM max. resolution is 10-bit
- ★ 10-bit multi-channel Analog-to-Digital converter
- ★ Synchronous Serial Port (SSP) with SPI™ (Master Mode) and I^2C™ (Master/Slave)
- ★ Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) 8-bits wide, with external $\overline{RD}$, $\overline{WR}$ and $\overline{CS}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

ANEXO B. Hoja de datos del Blueport

BRADATECH

Bluetooth

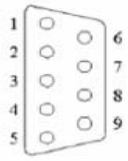
Bluetooth Serial Adapter

Product Specifications

Technical Specifications

Radio Transmission	
Frequency Range	2400MHz to 2483.5MHz
Transmit Power	Class 1 (max 20dBm)
Input Sensitivity	-84 to –20dBm
Modulation	GFSK, 1Mbps, 0.5BT 1600 hops/sec, 1MHz channel
Connectors	
RS232	SUB-D 9 Pin, Female or Male (DTE/DCE) Connector Available 1200 Baud to 230 kBaud Data Rates Baud rate, parity, data bits set using included configuration software
Antenna	Internal SMD
Power Supply	3.5 to 16 volt (includes 9V Battery Connector)
Bluetooth	
Version	1.1
Bluetooth Profiles	Generic Access, Service Discovery, Serial Port, SPP compatible
Device Role	Slave and/or Master
General	
Dimensions	45x60x12mm

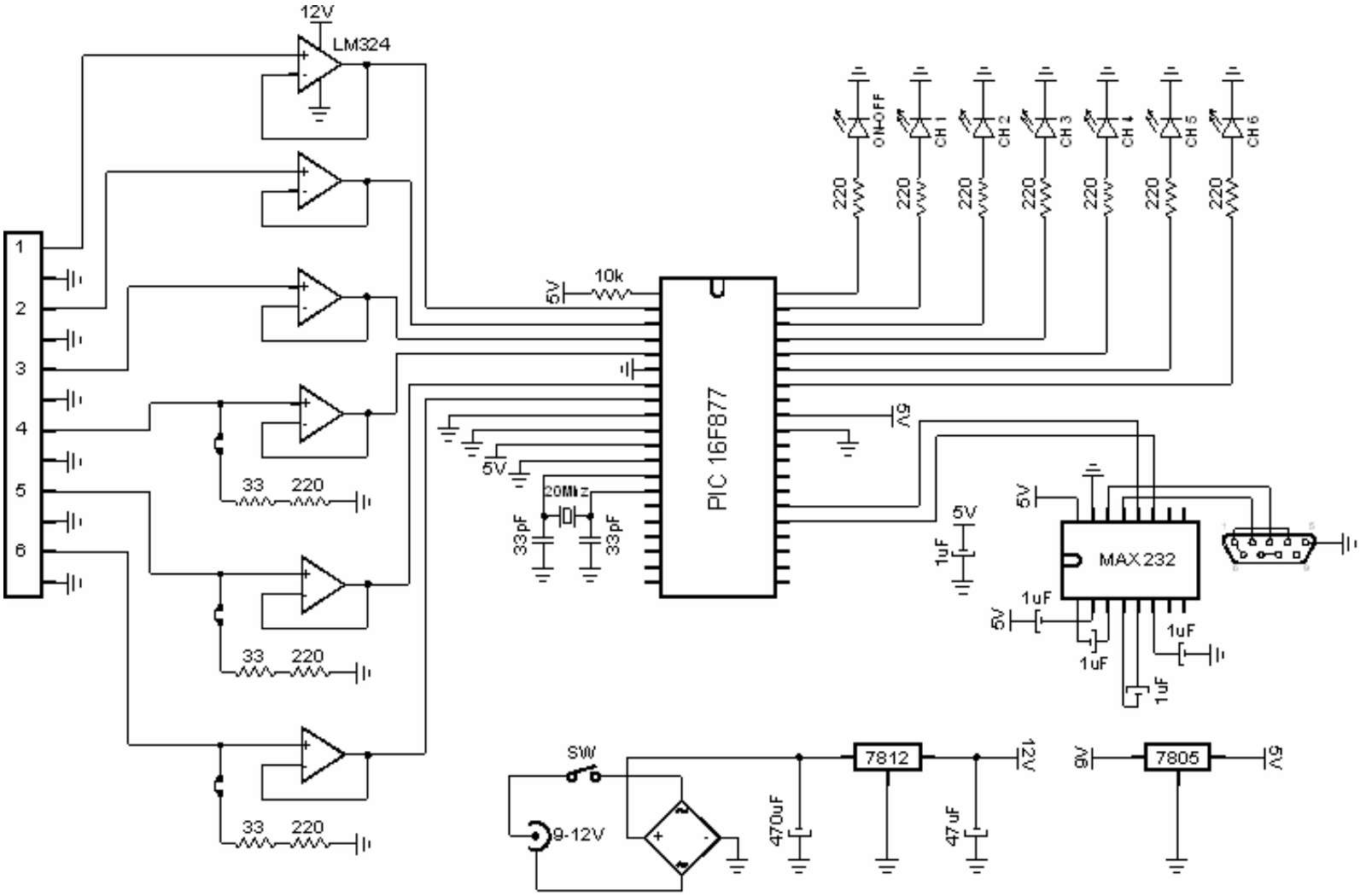
Serial Connector Pin Out



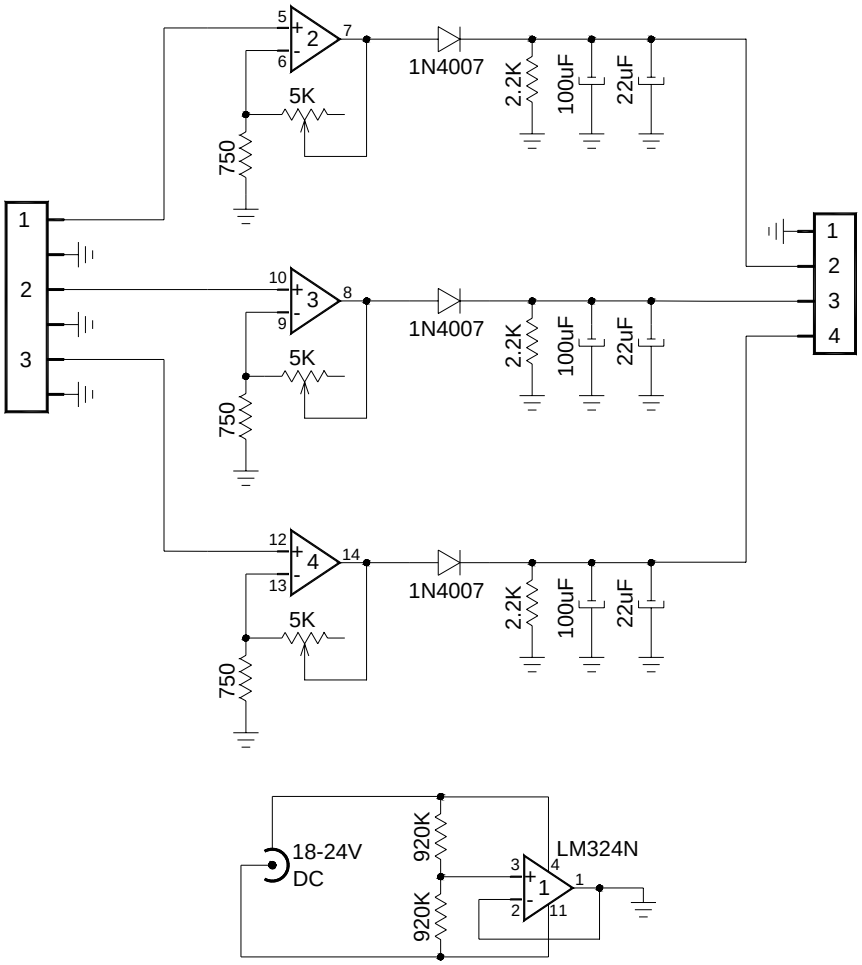
Pin no.	Signal name	Direction when connected to a DTE (PC)	Direction when connected to a DCE (Modem)
1	CD	no connection	no connection
2	RX	output	input
3	TX	input	output
4	DTR	no connection	no connection
5	Ground	-	-
6	DSR	no connection	no connection
7	RTS	input	output
8	CTS	output	input
9	RI	no connection	no connection

Support-Hotline 1-613-841-2295 * techsupport@bradatech.com

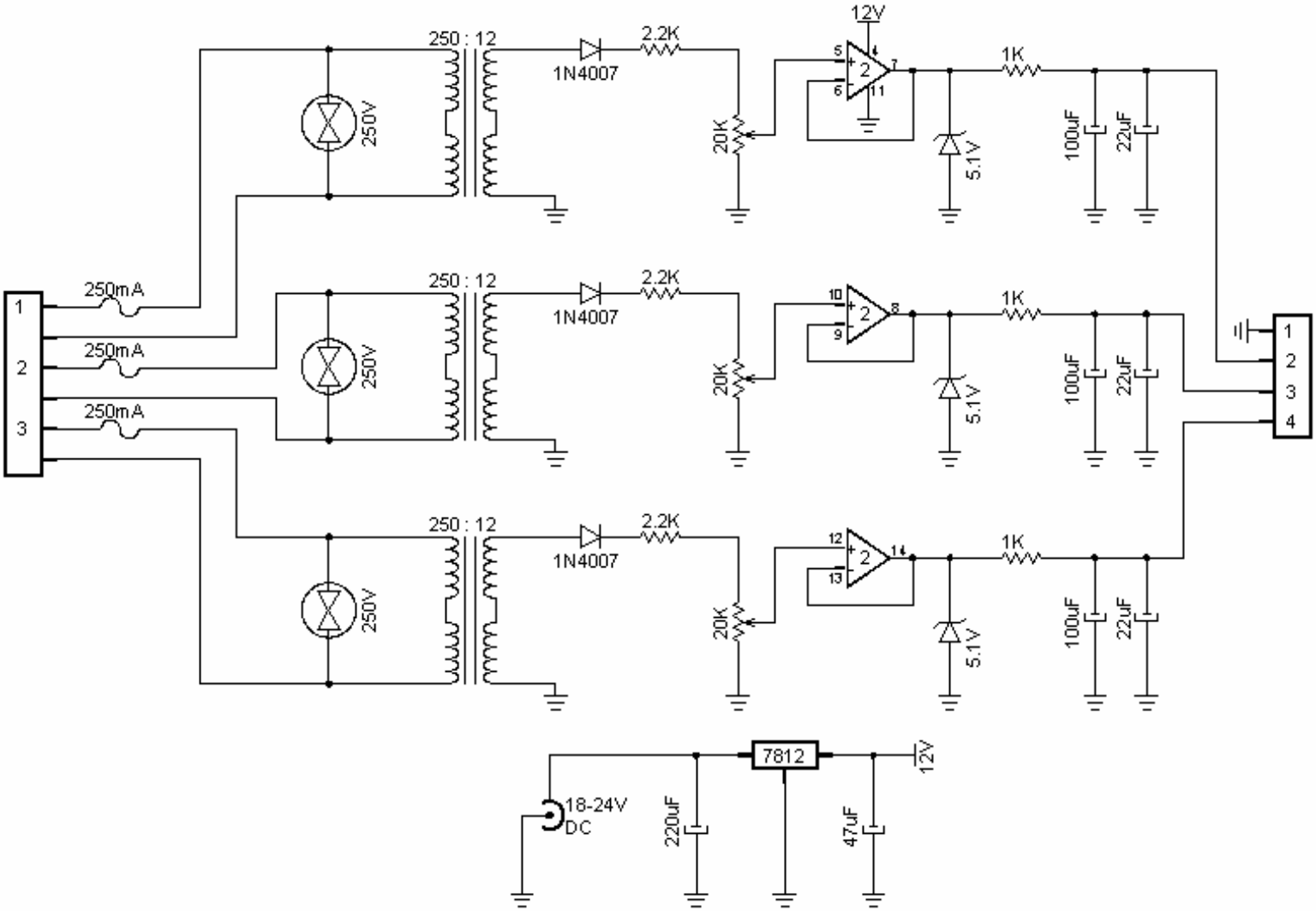
ANEXO C. Esquemático del módulo de adquisición de datos



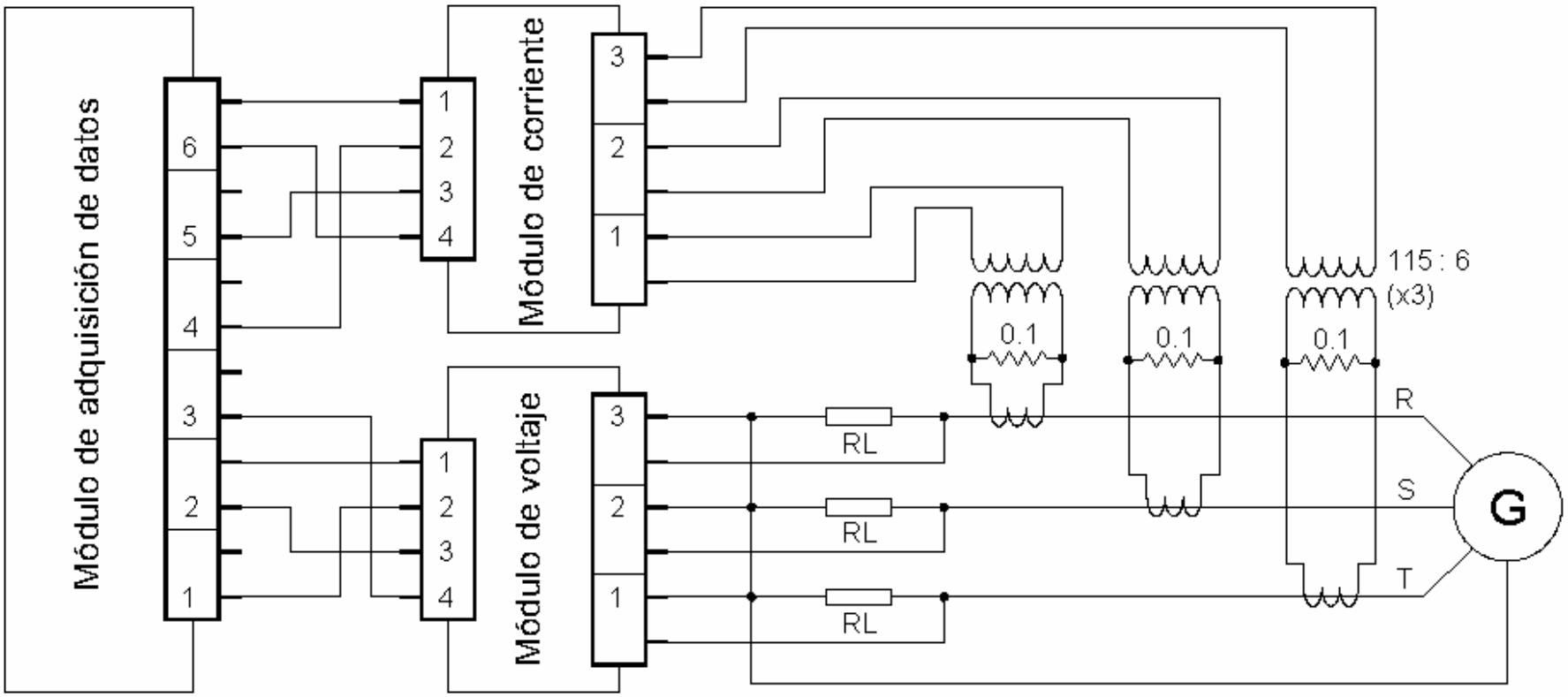
ANEXO D. Esquemático del módulo de corriente



ANEXO E. Esquemático del módulo de voltaje



ANEXO F. Montaje



ANEXO G. Código Fuente Microcontrolador

```
'*****************************************************************
'* Name    : SenSing.BAS                                         *
'* Author  : Delber Alberto Arzuza y Jose Ignacio Villareal      *
'* Notice  : Copyright (c) 2005                                  *
'*         : TODOS LOS DERECHOS RESERVADOS                       *
'* Date    : 15/03/2005                                          *
'* Version : 1.0                                                 *
'* Notes   :                                                     *
'*         :                                                     *
'*****************************************************************
DEFINE OSC 20         'se especifica la velocidad de oscilador
DEFINE ADC_BITS 10    'Se selecciona conversion de 10 Bits
DEFINE ADC_SAMPLEUS 50 'se utiliza un muestreo de 50uS
DEFINE ADC_CLOCK 3    'Se selecciona un periodo para el reloj de muestreo
'a través de un oscilador RC interna del procesador
TRISB = %00000000     'Todo el puerto B como salida para los LEDS
TRISA = %11111111     'Todo el puerto A como entrada para la conversion A/D
TRISC = %10000000     'Todo el puerto C como salidas excepto C7 que es Rx
TRISD = %00000000     'Todo el puerto D como salida ya que no se usa
ADCON1 = %10001001    'Se utilizan todas las entradas analogas y
'justificación a la derecha
RCSTA = %10010000     'se enciende la USART y se coloca a trabajar en modo
'de recepcion continua y con formato 8N1
TXSTA = %00100100     'Se selecciona transmision Asincrona con formato 8N1
SPBRG = 64            'Se configura la velocidad de transmision (19.2 KBit/s)

CONFIG VAR BIT     'Variable para almacenar si el módulo ha sido configurado
TEMP    VAR WORD   'Variable temporal para calculos intermedios (de chequeo)
ENVIA   VAR WORD   'Variable temporal para enviar los datos de la conversion
IN      VAR WORD   'Variable temporal para recibir ordenes desde el PC
I       VAR BYTE   'Variable termpoal para realizar los bucles
EN_CANAL VAR BYTE[7]'Variables para indicar cuales fueron los canales
'habilitados en la configuración
CANAL   VAR BYTE   'Variable temporal para enviar el canal muestreado
```

```
ANTERIOR VAR BYTE   'Variable para almacenar el ultimo caracter recibido
TX      VAR PORTC.5  'Pin por el que la USART transmite
ON_OFF   VAR PORTB.7  'Led para indicar que el modulo esta encendido
PORTB = 0         'Inicialmente los leds estan apagados
PAUSE 100         'Tiempo de espera inicial
HIGH ON_OFF       'Prende el led de encendido
HIGH TX           'Colocacion del pin Tx en alto para cumplir con la
'forma adecuada de los niveles RS232 (linea desocupada
'siempre en alto)
TEMP = 0          'Inicializacion de la variable temporal
ANTERIOR = 0      'Inicialización de la variable
CONFIG = 0        'Inicialización para indicar que el módulo se encuentra sin
'configurar
ON INTERRUPT GOTO LEER  'Indica la etiqueta a la que va despues de la
'interrupcion
PIE1.5 = 1          'Habilita la interrupcion por recepcion de
'la USART
INTCON = %11000000     'Habilita la interrupcion de perifericos

'*********************************
ESPERA:                           '* ciclo de inactividad,
IF CONFIG == 1 THEN               '* enciende los leds de
FOR I = 1 TO 6                    '* los canales que se
If EN_CANAL[I] = 1 THEN           '* habilitan
PORTB.0[I] = 1                     '*
ENDIF                              '*
NEXT I                            '*
ENDIF                             '*
GOTO ESPERA                        '*
'*********************************

'************************
DISABLE                     '* subrutina para el
LEER:                       '* manejo de las
IN = RCREG                  '* interrupciones.
IF IN == 68 THEN             '*
FOR I = 1 TO 6              '*
```

```
EN_CANAL[I] = 0            '*
NEXT I                     '*
PORTB = %10000000          '*
CONFIG = 0                 '*
RESUME ESPERA              '*
ENDIF                      '*
RESUME ANALIZA             '*
ENABLE                     '*
'**************************

'*******************************
INICIO:                          '*
FOR I = 1 TO 6                   '*
IF (EN_CANAL[I] == 1) THEN       '*
ADCIN (I - 1), ENVIA             '*
PAUSEUS 500                      '*
CANAL = I                        '*
GOSUB MANDAR                     '*
ENDIF                            '*
NEXT I                           '*
FOR I = 1 TO 6                   '*
IF (EN_CANAL[I] == 1) THEN       '*
PORTB.0[I] = 1                   '*
PAUSE 50                         '*
PORTB.0[I] = 0                   '*
ENDIF                            '*
NEXT I                           '*
HSEROUT [13]                     '*
PAUSE 45                         '*
GOTO INICIO                      '*
'*******************************

'************************************************
MANDAR:                        '* subrutina que calcula
TEMP = (9999 - ENVIA)                '* en complemento a 9999
IF ENVIA > 999 then                '* del valor medido.
HSEROUT [#CANAL,#ENVIA,#TEMP]          '* transmision del canal,
```

```
ENDIF                          '* valor, complemento.
'*
IF (ENVIA > 99) AND (ENVIA <= 999) then     '*
HSEROUT [#CANAL,"0",#ENVIA,#TEMP]       '*
ENDIF                          '* los ceros adicionales
'* son para transmitir el
IF (ENVIA > 9) AND (ENVIA <= 99) then       '* valor medido en un
HSEROUT [#CANAL,"0","0",#ENVIA,#TEMP]    '* numero constante de
ENDIF                          '* 4 bytes
'*
IF ENVIA <= 9 then                   '*
HSEROUT [#CANAL,"0","0","0",#ENVIA,#TEMP] '*
ENDIF                          '*
RETURN                          '*
'************************************************

'*************************
ANALIZA:            '* analiza los comandos
IF IN == ANTERIOR THEN  '*
RESUME          '*
ENDIF             '*
ANTERIOR = IN        '*
IF IN == 73 THEN      '* del PC para realizar
RESUME INICIO     '* las acciones pertinentes
ENDIF             '*
SELECT CASE in       '*
CASE 49         '*
EN_CANAL[1] = 1 '*
CONFIG = 1    '*
CASE 50         '*
EN_CANAL[2] = 1 '*
CONFIG = 1    '*
CASE 51         '*
EN_CANAL[3] = 1 '*
CONFIG = 1    '*
CASE 52         '*
EN_CANAL[4] = 1 '*
```

```
CONFIG = 1    '*
CASE 53       '*
EN_CANAL[5] = 1 '*
CONFIG = 1    '*
CASE 54       '*
EN_CANAL[6] = 1 '*
CONFIG = 1    '*
CASE 55       '*
EN_CANAL[1] = 1 '*
EN_CANAL[2] = 1 '*
EN_CANAL[3] = 1 '*
EN_CANAL[4] = 1 '*
EN_CANAL[5] = 1 '*
EN_CANAL[6] = 1 '*
CONFIG = 1    '*
END SELECT        '*
RESUME           '*
ENABLE           '*
'*************************

END ' Fin de programa
```

ANEXO G. Código Fuente MATLAB

```
function []=SenSING()

%Sistema de Monitoreo Inalambrico.
%Ayuda en linea
%--------------------------------------------------------------------------
%Borra la pantalla y las variables del espacio de trabajo
clc;
clear all;
%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
%La estructura DATOS contiene todas las variables y todos los datos usados en la interfaz
global DATOS;
%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
%INICIALIZACION DE VARIABLES
DATOS.Inicio=0;
DATOS.ArchivoNuevo=0;
DATOS.Modo=0;

DATOS.Configuracion(1)=1;
DATOS.Configuracion(2)=6;
DATOS.Configuracion(3)=4;
DATOS.Configuracion(4)=3;
DATOS.Configuracion(5)=1;
DATOS.Configuracion(6)=3;

DATOS.paso=0;

DATOS.AsistenteEnab=0;

%Configuracion del puerto serial
DATOS.Puerto=serial('com1','baudrate',19200,'terminator','CR','bytesavailablefcnmode','termina
tor','bytesavailablefcn',@Recibir);

DATOS.Reiniciar=0;
```

```
DATOS.DetenerA=0;

%Vector para identificar los canales habilitados
for k=1:6
   DATOS.Bandera(k)=0;
end
%------------------------------------------------------------------------
%------------------------------------------------------------------------
%CREACION DE LA INTERFAZ GRAFICA DE USUARIO

%Figura principal 0.517 0.612 0.705
DATOS.h(1)=figure('position',[20 1000 1250 900],'units','pixels','color',[0 0
0],'menubar','none',...
   'name','SenSING V1.0','numbertitle','off','visible','off','deletefcn',@salir);

%Creacion de los ejes para la imagen en la figura principal
DATOS.ejes(7)=axes;
%Carga la imagen
panel = imread('Panel de control', 'bmp');
%Muestra la imagen
image(panel);
%Reconfiguracion de los ejes
set(DATOS.ejes(7),'Visible', 'off','Units', 'pixels','Position', [0 35 800 525 ]);

%Creacion de los ejes para las graficas
for k=1:6
   DATOS.ejesg(k)=axes;
   set(DATOS.ejesg(k),'Visible', 'off','Units', 'pixels','Position', [143 398 248 122 ]);
end

%Creacion de los Scroll's para el manejo de las graficas
for k=1:6
   DATOS.EsGrafica(k)=uicontrol('style','slider','backgroundcolor',[0.8 0.8
0.8],'min',0,'max',5000,...
      'sliderstep',[0.0002 0.002],'enable','off');
end

%Creacion de los cuadros de texto para las lecturas digitales
for k=1:6
   DATOS.TeLectura(k)=uicontrol('style','text','string','','fontsize',15,...
```

```
    'backgroundcolor',[0 0 0],'foregroundcolor',[1 1 0],'horizontalalignment','right');
end

%Creacion de los cuadros de texto para la variable de entrada
for k=1:6
   DATOS.TeCanal(k)=uicontrol('style','text','backgroundcolor',[0 0 0],...
      'foregroundcolor',[1 1 0],'horizontalalignment','left');
end

%Inicia con la pantalla minimizada
DATOS.estadopantalla=0;

%CREACION DE LA BARRA DE MENU
%Menu Archivo
DATOS.MeArchivo=uimenu('Label','&Archivo');
   DATOS.MeAbrir=uimenu(DATOS.MeArchivo,'Label','Abrir','accelerator','A','callback',@abrir);

DATOS.MeNuevo=uimenu(DATOS.MeArchivo,'Label','Nuevo','accelerator','N','callback',@nuevo);

DATOS.MeSalir=uimenu(DATOS.MeArchivo,'Label','Salir','callback',@salir,'accelerator','S','separat
or','on');

%Menu Opciones
DATOS.MeOpciones=uimenu('Label','&Opciones','enable','off');
   DATOS.MeCanales=uimenu(DATOS.MeOpciones,'Label','Canales','accelerator','C');

DATOS.MeDefecto=uimenu(DATOS.MeCanales,'Label','Defecto','accelerator','D','callback',@Defect
o);

DATOS.MeVoltaje=uimenu(DATOS.MeCanales,'Label','Voltaje','accelerator','V','callback',@CanalV
oltaje,'separator','on');

DATOS.MeCorriente=uimenu(DATOS.MeCanales,'Label','Corriente','accelerator','A','callback',@Ca
nalCorriente);

DATOS.MeIniciar=uimenu(DATOS.MeOpciones,'Label','Iniciar','accelerator','I','separator','on',...
      'callback',@Iniciar,'enable','off');

DATOS.MeDetener=uimenu(DATOS.MeOpciones,'Label','Detener','accelerator','D','callback',@Dete
ner,...
      'enable','off');

%Menu Herramientas
DATOS.MeHerramientas=uimenu('Label','&Herramientas');
```

```
   DATOS.MeConfigurar=uimenu(DATOS.MeHerramientas,'Label','Configurar
puerto','accelerator','C','callback',@ConfigurarPuerto);
   %DATOS.MeBluetooth=uimenu(DATOS.MeHerramientas,'Label','Asistente
Bluetooth','accelerator','B','callback',@Asistente,'separator','on');


%Menu Ayuda
DATOS.MeAyuda=uimenu('Label','&Ayuda');
   DATOS.MeHelp=uimenu(DATOS.MeAyuda,'Label','Ayuda','accelerator','H','callback',@Ayuda);
   DATOS.MeAcercaDe=uimenu(DATOS.MeAyuda,'Label','Acerca de
SenSING','separator','on','callback',@acerca_de,...
      'accelerator','A');


%Presenta la figura y la define como restaurada
set(DATOS.h(1),'visible','on','resizefcn',@maximizar);


%Invoca la funcion nuevo
nuevo


%Invoca la funcion acerca_de
acerca_de
%--------------------------------------------------------------------


%FUNCIONES DEL MENU
%--------------------------------------------------------------------
function  [varargout]=abrir(varargin)


%Definicion de variables
global DATOS;


DATOS.RutaNombre='';


%Presenta la ventana para obtener la ruta y el nombre del archivo
[DATOS.filename, DATOS.pathname] = uigetfile({'*.xls','Archivos de Microsoft Excel (*.xls)'},
'Abrir');
%Abre el archivo
if ~isequal(DATOS.pathname,0) & ~isequal(DATOS.filename,0)
   DATOS.RutaNombre=strcat(DATOS.pathname, DATOS.filename);
end


if  ~isequal(DATOS.RutaNombre,'')
   DATOS.matriz=xlsread(DATOS.RutaNombre);
```

```
%Obtiene las dimensiones de la matriz
[filas columnas]=size(DATOS.matriz);
%Crea el vecto de tiempo
x=1:1:filas;

for k=1:6
   %Borra las graficas anteriores
   axes(DATOS.ejesg(k));
   DATOS.grafica(k)=plot(0,0);
   set(DATOS.grafica(k),'color',[1 1 0],'xdata',[],'ydata',[]);
   set(DATOS.ejesg(k),'visible','off','drawmode','fast');
   %Deshabilita todos los escroll
   set(DATOS.EsGrafica(k),'enable','off');
   %Borra todos los cuadros para las lecturas
   set(DATOS.TeLectura(k),'string','');
   %Vector para identificar los canales habilitados
   for k=1:6
      DATOS.Bandera(k)=0;
   end
end

%Traza las nuevas graficas
for k=1:columnas
   %Comprueba si hay datos para cada canal
   if isnan(DATOS.matriz(1,k))~=1
      %Carga la grafica
      axes(DATOS.ejesg(k));
      DATOS.grafica(k)=plot(x,DATOS.matriz(:,k));
      set(DATOS.grafica(k),'color',[1 1 0]);
      axis([0 62 min(DATOS.matriz(:,k))-0.2 max(DATOS.matriz(:,k))+0.2]);
      set(DATOS.ejesg(k),'visible','off','drawmode','fast');
      %Habilita los scroll y la funcion
      set(DATOS.EsGrafica(k),'enable','on','callback',@scroll);
      if filas<=62
         set(DATOS.EsGrafica(k),'value',columnas);
      else
         set(DATOS.EsGrafica(k),'value',62);
      end
      %Actualiza los cuadros para las lecturas
```

```
        actual=get(DATOS.EsGrafica(k),'value');
        set(DATOS.TeLectura(k),'string',num2str(DATOS.matriz(actual,k),'%3.2f'));
        %Identifica los canales utilizados
        DATOS.Bandera(k)=1;
      end
    end

    %Guarda los valores iniciales
    DATOS.InicioScroll=get(DATOS.EsGrafica,'value');
    DATOS.InicioScroll=cell2mat(DATOS.InicioScroll);

    %Actualiza el modo a viejo
    DATOS.Modo='Viejo';
    %Desactiva el menu Opciones
    set(DATOS.MeOpciones,'enable','off');

    %Deshabilita la opcion Nuevo
    set(DATOS.MeNuevo,'enable','off');
end
%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
function [varargout]=scroll(varargin)

global DATOS

%Obtiene los datos actuales de los scroll
DATOS.NuevoScroll=get(DATOS.EsGrafica,'value');
DATOS.NuevoScroll=cell2mat(DATOS.NuevoScroll);

%Obtiene las dimensiones de la matriz
[filas columnas]=size(DATOS.matriz);

%Crea el vecto de tiempo
x=1:1:filas;

%Actualiza la grafica
for k=1:6
  if DATOS.Bandera(k)==1
```

```
      if ~isequal(DATOS.NuevoScroll(k),DATOS.InicioScroll(k))
         if DATOS.NuevoScroll(k)<filas & DATOS.NuevoScroll(k)>62
            if isnan(DATOS.matriz(DATOS.NuevoScroll(k),k))~=1
               %Carga la grafica
               axes(DATOS.ejesg(k));
               xlim([DATOS.NuevoScroll(k)-62,DATOS.NuevoScroll(k)]);
               %set(DATOS.ejesg(k),'visible','off','drawmode','fast');
               set(DATOS.grafica(k),'xdata',DATOS.NuevoScroll(k)-
61:1:DATOS.NuevoScroll(k),'ydata',DATOS.matriz(DATOS.NuevoScroll(k)-
61:DATOS.NuevoScroll(k),k));
               %Actualiza los cuadros para las lecturas

set(DATOS.TeLectura(k),'string',num2str(DATOS.matriz(DATOS.NuevoScroll(k),k),'%3.2f'));
            end
         end
         if DATOS.NuevoScroll(k)<=62 & DATOS.NuevoScroll(k)>0
            if isnan(DATOS.matriz(DATOS.NuevoScroll(k),k))~=1
               %Carga la grafica
               axes(DATOS.ejesg(k));
               xlim([DATOS.NuevoScroll(k)-62,DATOS.NuevoScroll(k)]);

set(DATOS.TeLectura(k),'string',num2str(DATOS.matriz(DATOS.NuevoScroll(k),k),'%3.2f'));
            end
         end
      end
   end
end

%Actualiza los datos
DATOS.InicioScroll=DATOS.NuevoScroll;
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=nuevo(varargin)

%Definicion de variables
global DATOS;

DATOS.RutaNombre='';

%Presenta la ventana para crear el archivo
if DATOS.ArchivoNuevo==1
```

```
    %Actualiza el modo a Nuevo
    DATOS.Modo='Nuevo';
    %Obtiene la ruta y el nombre del archivo
    [DATOS.filename, DATOS.pathname] = uiputfile({'*.xls','Archivos de Microsoft Excel
(*.xls)'},'Guardar como');

    %Crea y abre el archivo
    if ~isequal(DATOS.pathname,0) & ~isequal(DATOS.filename,0)
        DATOS.RutaNombre=strcat(DATOS.pathname, DATOS.filename);

        %Agrega la extension al nombre del archivo
        if isequal(findstr(DATOS.RutaNombre,'.xls'),[])
            DATOS.RutaNombre=strcat(DATOS.RutaNombre,'.xls');
        end

        %Inicia un servidor para Excel
        DATOS.e = actxserver('excel.application');
        set(DATOS.e,'windowstate','xlminimized')
        %Inserta un nuevo libro
        eWorkbooks = get(DATOS.e, 'Workbooks');
        DATOS.eWorkbook = Add(eWorkbooks);
        set(DATOS.e, 'Visible', 1);
        %Activa la primera hoja
        eActiveWorkbook = get(DATOS.e, 'ActiveWorkBook');
        eSheets = get(eActiveWorkbook, 'Sheets');
        eSheet1 = Item(eSheets, 1);
        Activate(eSheet1);
        %Optiene un manejador para la hoja activa
        DATOS.eActiveSheet = get(DATOS.e, 'ActiveSheet');
        % Guarda el libro
        SaveAs(DATOS.eWorkbook, DATOS.RutaNombre);

        set(DATOS.MeOpciones,'enable','on');
        %set(DATOS.MeCorriente,'enable','off');
        DATOS.ConfigurarCorriente=0;
        %Numero de canales habilitados
        DATOS.Cantidad=0;

        %Deshabilita las opciones
        set(DATOS.MeAbrir,'enable','off');
```

```
      set(DATOS.MeNuevo,'enable','off');
   end
end

%Inicializacion
for k=1:3
   DATOS.vc{k,1}='Voltaje [V]';
   DATOS.vc{k+3,1}='Corriente [mA]';
end

for k=1:3
   DATOS.ValMinC(k)=0;
   DATOS.ValMaxC(k)=150;
   DATOS.VisMinC(k)=0;
   DATOS.VisMaxC(k)=150;
   DATOS.Estados(k)=0;
end

for k=4:6
   DATOS.ValMinC(k)=0;
   DATOS.ValMaxC(k)=1.4;
   DATOS.VisMinC(k)=0;
   DATOS.VisMaxC(k)=1.4;
   DATOS.Estados(k)=0;
end

%Inicializacion de la funcion ArchivoNuevo
DATOS.ArchivoNuevo=1;
%--------------------------------------------------------------------------

%--------------------------------------------------------------------------
function [varargout]=salir(varargin)

%Variables utilizadas por la funcion
global DATOS;

%Cierra el puerto
if strcmp(DATOS.Puerto.status,'open')
   stopasync(DATOS.Puerto);
```

```
   fclose(DATOS.Puerto);
end

%Cierra la aplicacion
delete(DATOS.h(1));

%Libera el recurso
fclose(DATOS.Puerto);;
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
function [varargout]=CanalVoltaje(varargin)

%Variables
global DATOS;

if DATOS.ConfigurarCorriente==0
   DATOS.Actual=1;
else
   DATOS.Actual=4;
end

%CREACION DE LA INTERFAZ
%Crea la figura y la oculta mientras la configura
DATOS.cv=figure('visible','off');
%Configuracion de la figura
set(DATOS.cv,'position',[150 165 500 250]);
set(DATOS.cv,'menubar','none','name','Canales 0-5 [V]','numbertitle','off','windowstyle','modal');
set(DATOS.cv,'resize','off','visible','on', 'Units', 'pixels');
set(DATOS.cv,'deletefcn',@Salir);

%Creacion de los controles
uicontrol('style','frame','position',[1 1 500 250]);
uicontrol('style','frame','position',[10 50 480 190]);
uicontrol('style','frame','position',[30 90 210 90]);
uicontrol('style','frame','position',[260 90 210 90]);

%Control para presentar el numero del canal
DATOS.numcanal=uicontrol('style','text','position',[15 230 40 15],'string','Canal 1');
```

```
uicontrol('style','text','position',[30 205 50 15],'string','Variable:','horizontalalignment','left');
uicontrol('style','text','position',[35 172 30 15],'string','Valor');
uicontrol('style','text','position',[265 172 70 15],'string','Visualizacion');
uicontrol('style','text','position',[40 142 40 15],'string','Minimo:','horizontalalignment','left');
uicontrol('style','text','position',[270 142 40 15],'string','Minimo:','horizontalalignment','left');
uicontrol('style','text','position',[40 112 40 15],'string','Maximo:','horizontalalignment','left');
uicontrol('style','text','position',[270 112 40 15],'string','Maximo:','horizontalalignment','left');


%Control para pedir el nombre de la variable
DATOS.nomvar=uicontrol('style','edit','position',[80 203 160
20],'horizontalalignment','left','backgroundcolor',[1 1 1],...
   'tooltipstring','Nombre de la variable que desea monitorear');


%Controles para los valores maximo y minimo de la variable medida
DATOS.varmin=uicontrol('style','edit','position',[90 140 130
20],'horizontalalignment','left','backgroundcolor',[1 1 1],...
   'tooltipstring','Valor minimo de la variable en el proceso');
DATOS.varmax=uicontrol('style','edit','position',[90 110 130
20],'horizontalalignment','left','backgroundcolor',[1 1 1],...
   'tooltipstring','Valor maximo de la variable en el proceso');


%Controles para los valores maximo y minimo de los ejes
DATOS.vismin=uicontrol('style','edit','position',[320 140 130
20],'horizontalalignment','left','backgroundcolor',[1 1 1],...
   'tooltipstring','Minimo valor de la variable que desea visualizar');
DATOS.vismax=uicontrol('style','edit','position',[320 110 130
20],'horizontalalignment','left','backgroundcolor',[1 1 1],...
   'tooltipstring','Maximo valor de la variable que desea visualizar');


%Control para habilitar canal
DATOS.habilitar(DATOS.Actual)=uicontrol('style','pushbutton','position',[390 60 80
20],'string','Aceptar','callback',@Habilitar);


%Botones
%DATOS.atras=uicontrol('style','pushbutton','position',[205 15 80 20],'string','<
Atras','enable','off');
DATOS.siguiente=uicontrol('style','pushbutton','position',[290 15 80 20],'string','Siguiente
>','enable','off','callback',@Siguiente);
DATOS.salir=uicontrol('style','pushbutton','position',[410 15 80
20],'string','Salir','callback',@Salir);


%CARGA DE VALORES ACTUALES
ActualizarCanal
```

```
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=Salir(varargin)

%Definicion de variables
global DATOS;

%Borra la interfaz
delete(DATOS.cv);

if DATOS.ConfigurarCorriente==0
   if DATOS.Bandera(1)==1|DATOS.Bandera(2)==1|DATOS.Bandera(3)==1
      set(DATOS.MeVoltaje,'enable','off');
      set(DATOS.MeDefecto,'enable','off');
      set(DATOS.MeCorriente,'enable','on');
   end
else
   if DATOS.Bandera(4)==1|DATOS.Bandera(5)==1|DATOS.Bandera(6)==1
      set(DATOS.MeDefecto,'enable','off');
      set(DATOS.MeCorriente,'enable','off');
   end
end

%Habilita la funcion Iniciar
if isequal(get(DATOS.MeVoltaje,'enable'),'off') & isequal(get(DATOS.MeCorriente,'enable'),'off')
   set(DATOS.MeIniciar,'enable','on');
   set(DATOS.MeCanales,'enable','off');
end
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=Siguiente(varargin)

%Definicion de variables
global DATOS;

if DATOS.ConfigurarCorriente==0
   if DATOS.Actual<3
      DATOS.Actual=DATOS.Actual+1;
      set(DATOS.siguiente,'enable','off');
```

```
        ActualizarCanal
    end
else
    if DATOS.Actual<6
        DATOS.Actual=DATOS.Actual+1;
        set(DATOS.siguiente,'enable','off');
        ActualizarCanal
    end
end
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=CanalCorriente(varargin)

%Definicion de variables
global DATOS;

%Definicion de una bandera
DATOS.ConfigurarCorriente=1;

%Carga la interfaz
CanalVoltaje
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=Iniciar(varargin)

%Variables utilizadas por la funcion
global DATOS;

    fopen(DATOS.Puerto);

    for k=1:6
        DATOS.x(k)=0;
    end

    fprintf(DATOS.Puerto,'%s','I','async');

    DATOS.Inicio=1;

    %Deshabilita la funcion Iniciar
```

```
    set(DATOS.MeIniciar,'enable','off');
    %Deshabilita el menu Herramientas
    set(DATOS.MeHerramientas,'enable','off');
    %Habilita el menu Detener
    set(DATOS.MeDetener,'enable','on');
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=Detener(varargin)

%Variables utilizadas por la funcion
global DATOS;

DATOS.DetenerA=1;
%Inicia el temporizador
DATOS.t2 = timer('TimerFcn',@Recibir, 'StartDelay',1);
start(DATOS.t2);
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=ConfigurarPuerto(varargin)

%Variables
global DATOS;

%CREACION DE LA INTERFAZ
%Crea la figura y la oculta mientras la configura
DATOS.Configurar=figure('visible','off');
%Configuracion de la figura
set(DATOS.Configurar,'position',[206 145 388 376]);

%Creacion de los ejes para la imagen en la figura principal
DATOS.ejesConf=axes;
%Carga la imagen
panel2 = imread('Configuracion', 'png');
%Muestra la imagen
image(panel2);
%Reconfiguracion de los ejes
set(DATOS.ejesConf,'Visible', 'off','Units', 'pixels','Position', [0 0 388 376 ]);
set(DATOS.Configurar,'menubar','none','name','Propiedades del
puerto','numbertitle','off','windowstyle','modal');
set(DATOS.Configurar,'resize','off','visible','on', 'Units', 'pixels');
```

```
%Creacion de los controles
uicontrol('style','text','position',[35 290 100 15],'string','Conectar
usando:','horizontalalignment','right','backgroundcolor',[0.9725 0.9725 0.9725]);
uicontrol('style','text','position',[35 254 100 15],'string','Bits por
segundo:','horizontalalignment','right','backgroundcolor',[0.9725 0.9725 0.9725]);
uicontrol('style','text','position',[35 218 100 15],'string','Bits de
datos:','horizontalalignment','right','backgroundcolor',[0.9725 0.9725 0.9725]);
uicontrol('style','text','position',[35 182 100
15],'string','Paridad:','horizontalalignment','right','backgroundcolor',[0.9725 0.9725 0.9725]);
uicontrol('style','text','position',[35 146 100 15],'string','Bits de
parada:','horizontalalignment','right','backgroundcolor',[0.9725 0.9725 0.9725]);
uicontrol('style','text','position',[35 110 100 15],'string','Control de
flujo:','horizontalalignment','right','backgroundcolor',[0.9725 0.9725 0.9725]);


%Controles para la entrada de datos
DATOS.CoBPOR=uicontrol('style','popupmenu','position',[150 293 140
15],'string','COM1|COM2|COM3|COM4|COM5|COM6','backgroundcolor',[1 1 1]);
DATOS.CoBPS=uicontrol('style','popupmenu','position',[150 257 140 15],...
   'string','300|1200|2400|4800|9600|19200|38400|57600|115200','backgroundcolor',[1 1 1]);
DATOS.CoBDAT=uicontrol('style','popupmenu','position',[150 221 140
15],'string','5|6|7|8','backgroundcolor',[1 1 1]);
DATOS.CoBP=uicontrol('style','popupmenu','position',[150 185 140
15],'string','Par|Impar|Ninguna|Marca|Espacio','backgroundcolor',[1 1 1]);
DATOS.CoBS=uicontrol('style','popupmenu','position',[150 149 140
15],'string','1|2','backgroundcolor',[1 1 1]);
DATOS.CoFLU=uicontrol('style','popupmenu','position',[150 113 140
15],'string','Xon/Xoff|Hardware|Ninguno','backgroundcolor',[1 1 1]);


%Botones
DATOS.aceptar=uicontrol('style','pushbutton','position',[207 10 80
22],'string','Aceptar','enable','on','callback',@ReConfigurarPuerto);
DATOS.cancelar=uicontrol('style','pushbutton','position',[300 10 80
22],'string','Cancelar','enable','on','callback',...
   ['global DATOS;','delete(DATOS.Configurar);']);
DATOS.restaurar=uicontrol('style','pushbutton','position',[197 55 165 22],'string','Restaurar
predeterminados','enable','on','callback',...
   ['global
DATOS;','set(DATOS.CoBPOR,''value'',1);','set(DATOS.CoBPS,''value'',6);','set(DATOS.CoBDAT,''va
lue'',4);',...
   'set(DATOS.CoBP,''value'',3);','set(DATOS.CoBS,''value'',1);','set(DATOS.CoFLU,''value'',3);']);


%Carga los valores actuales
set(DATOS.CoBPOR,'value',DATOS.Configuracion(1));
set(DATOS.CoBPS,'value',DATOS.Configuracion(2));
set(DATOS.CoBDAT,'value',DATOS.Configuracion(3));
```

```
set(DATOS.CoBP,'value',DATOS.Configuracion(4));
set(DATOS.CoBS,'value',DATOS.Configuracion(5));
set(DATOS.CoFLU,'value',DATOS.Configuracion(6));
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=Asistente(varargin)

%Definicion de variables
global DATOS;

%Solo carga la pantalla principal cuando se llama el Asistente
if DATOS.AsistenteEnab==0
   %Establece el paso actual del asistente
   DATOS.paso=0;
   %Figura principal
   DATOS.h(3)=figure('position',[100 140 600 300],'units','pixels','menubar','none',...
      'name','Asistente de configuración del BluePort','numbertitle','off','visible','off',...
      'deletefcn',['global
DATOS;','DATOS.paso=0;','DATOS.AsistenteEnab=0;'],'windowstyle','modal','resize','off');

   %Creacion de los ejes para la imagen en la figura principal
   DATOS.ejes(9)=axes;
   %Carga la imagen
   panel = imread('Asistente', 'png');
   %Muestra la imagen
   image(panel);
   %Reconfiguracion de los ejes
   set(DATOS.ejes(9),'Visible', 'off','Units', 'pixels','Position', [0 0 600 300]);

    %Presenta la figura
   set(DATOS.h(3),'visible','on');

   %Boton Siguiente
   DATOS.siguienteAsist=uicontrol('style','pushbutton','position',[380 10 80
22],'string','Siguiente >','enable','on',...
      'callback',@SigPasoAsistente);
end

%Señala el estado de ejecucion del asistente
DATOS.AsistenteEnab=1;
```

```
switch DATOS.paso
   case 0
      %Presenta la primera pantalla
      DATOS.PasosAsistente(1)=uicontrol('style','text','position',[160 250 420
30],'horizontalalignment','left',...
         'backgroundcolor',[0.9255 0.9137 0.8471],...
         'string','Con el Asistente Bluetooth de SenSING podrá configurar el adaptador de
comunicaciones Blueport de forma rápida y sencilla.');

      DATOS.PasosAsistente(2)=uicontrol('style','text','position',[160 210 420
30],'horizontalalignment','left',...
         'backgroundcolor',[0.9255 0.9137 0.8471],...
         'string','El Asistente le permite seleccionar entre diversas opciones de configuración para
la comunicación serial; para ello, siga cada uno de los pasos indicados.');

   case 1
      %Presenta la segunda pantalla
      DATOS.PasosAsistente(1)=uicontrol('style','text','position',[160 240 420
40],'horizontalalignment','left',...
         'backgroundcolor',[0.9255 0.9137 0.8471],...
         'string','PASO 1:
Conecte el Blueport en el puerto COM1 y luego presione el botón Examinar para reconocer el
dispositivo.');
      DATOS.PasosAsistente(2)=uicontrol('style','text','position',[160 80 420
20],'horizontalalignment','left',...
         'backgroundcolor',[0.9255 0.9137 0.8471],...
         'string','Nombre del dispositivo:');
      DATOS.PasosAsistente(3)=uicontrol('style','text','position',[160 65 320
15],'horizontalalignment','left',...
         'backgroundcolor',[1 1 1],...
         'string','');
      DATOS.PasosAsistente(4)=uicontrol('style','pushbutton','position',[502 62 80
22],'string','Examinar...',...
         'callback',['global
DATOS;','fopen(DATOS.Puerto);','fprintf(DATOS.Puerto,''?F0'');','DATOS.t1=timer(''TimerFcn'',''fc
lose(DATOS.Puerto)'', ''StartDelay'',1);','start(DATOS.t1);']);

   case 2
      %Presenta la tercera pantalla
      DATOS.PasosAsistente(1)=uicontrol('style','text','position',[160 240 420
40],'horizontalalignment','left',...
         'backgroundcolor',[0.9255 0.9137 0.8471],...
         'string','PASO 2:
Ingrese los nuevos parametros para la configuracion del Blueport (comunicacion serial).');
```

```
      DATOS.PasosAsistente(2)=uicontrol('style','text','position',[150 204 150 15],'string','Bits
por segundo:','horizontalalignment','right');

      DATOS.PasosAsistente(3)=uicontrol('style','text','position',[150 168 150 15],'string','Bits de
datos:','horizontalalignment','right');

      DATOS.PasosAsistente(4)=uicontrol('style','text','position',[150 132 150
15],'string','Paridad:','horizontalalignment','right');

      DATOS.PasosAsistente(5)=uicontrol('style','text','position',[150 96 150 15],'string','Bits de
parada:','horizontalalignment','right');

      DATOS.PasosAsistente(6)=uicontrol('style','text','position',[150 60 150 15],'string','Control
de flujo:','horizontalalignment','right');


      DATOS.BlueConf(1)=uicontrol('style','popupmenu','position',[310 207 140 15],...

         'string','19200|38400|57600|115200','backgroundcolor',[1 1 1]);

      DATOS.BlueConf(2)=uicontrol('style','popupmenu','position',[310 171 140
15],'string','7|8','backgroundcolor',[1 1 1]);

      DATOS.BlueConf(3)=uicontrol('style','popupmenu','position',[310 135 140
15],'string','Par|Impar|Ninguna','backgroundcolor',[1 1 1]);

      DATOS.BlueConf(4)=uicontrol('style','popupmenu','position',[310 99 140
15],'string','1|2','backgroundcolor',[1 1 1]);

      DATOS.BlueConf(5)=uicontrol('style','popupmenu','position',[310 63 140
15],'string','Hardware|Ninguno','backgroundcolor',[1 1 1]);


      DATOS.PasosAsistente(7)=uicontrol('style','pushbutton','position',[502 57 80
22],'string','Aplicar',...

        'callback',@ReConfigurarBluePort1);


      %Carga los valores actuales

      set(DATOS.BlueConf(1),'value',1);

      set(DATOS.BlueConf(2),'value',2);

      set(DATOS.BlueConf(3),'value',3);

      set(DATOS.BlueConf(5),'value',2);

   case 3

      %Presenta la pantalla final

      DATOS.PasosAsistente(1)=uicontrol('style','text','position',[160 240 420
40],'horizontalalignment','left',...

         'backgroundcolor',[0.9255 0.9137 0.8471],...

        'string','PASO 3:
Ingrese los nuevos parametros para la configuracion del Blueport (acceso y presentacion).');


      DATOS.PasosAsistente(2)=uicontrol('style','text','position',[150 204 150
15],'string','Nombre del dispositivo:','horizontalalignment','right');

      DATOS.PasosAsistente(3)=uicontrol('style','text','position',[150 168 150
15],'string','Nombre del servicio:','horizontalalignment','right');
```

```
    DATOS.PasosAsistente(4)=uicontrol('style','text','position',[150 132 150 15],'string','Codigo
PIN:','horizontalalignment','right');

    %DATOS.PasosAsistente(5)=uicontrol('style','text','position',[150 96 150
15],'string','Direcciones permitidas:','horizontalalignment','right');

    DATOS.PasosAsistente(6)=uicontrol('style','text','position',[150 96 150
15],'string','Encriptacion/Autenticacion:','horizontalalignment','right');


    DATOS.BlueConf(6)=uicontrol('style','edit','position',[310 202 220 20],'backgroundcolor',[1
1 1],'horizontalalignment','left');

    DATOS.BlueConf(7)=uicontrol('style','edit','position',[310 166 220 20],'backgroundcolor',[1
1 1],'horizontalalignment','left');

    DATOS.BlueConf(8)=uicontrol('style','edit','position',[310 130 220 20],'backgroundcolor',[1
1 1],'horizontalalignment','left');

    %DATOS.BlueConf(9)=uicontrol('style','edit','position',[310 99 220 15],'backgroundcolor',[1
1 1],'horizontalalignment','left');

    DATOS.BlueConf(10)=uicontrol('style','popupmenu','position',[310 99 220 15],...

      'string','Deshabilitar Encriptacion y Autenticacion|Habilitar Encriptacion|Habilitar
Autenticacion|Habilitar Encriptacion y Autenticacion',...

      'backgroundcolor',[1 1 1],'horizontalalignment','left');


    %Lee los valores actuales

    DATOS.set=0;

    DATOS.Leido=0;

    ReConfigurarBluePort2;


end


%Siempre presenta los botones

DATOS.salirAsist=uicontrol('style','pushbutton','position',[502 10 80
22],'string','Cancelar','enable','on',...

    'callback',['global DATOS;',
'delete(DATOS.h(3));','DATOS.paso=0;','DATOS.AsistenteEnab=0;']);

%------------------------------------------------------------------------

%------------------------------------------------------------------------

function [varargout]=SigPasoAsistente(varargin)


%Definicion de variables

global DATOS;


%Borra los controles anteriores

switch DATOS.paso

  case 0

    delete(DATOS.PasosAsistente(1));
```

```
      delete(DATOS.PasosAsistente(2));
      set(DATOS.siguienteAsist,'enable','off');
   case 1
      for k=1:4
         delete(DATOS.PasosAsistente(k));
      end
   case 2
      for k=1:7
         delete(DATOS.PasosAsistente(k));
      end
      for k=1:5
         delete(DATOS.BlueConf(k));
      end
      set(DATOS.siguienteAsist,'string','Finalizar');
  end

DATOS.paso=DATOS.paso+1;

if DATOS.paso<4
   Asistente
else
   DATOS.set=1;
   ReConfigurarBluePort2;
end
%--------------------------------------------------------------------
%--------------------------------------------------------------------
function [varargout]=Ayuda(varargin)
open('sensing.htm');
%--------------------------------------------------------------------
%--------------------------------------------------------------------
%Presentacion de la pantalla acerca_de
function [varargout]=acerca_de(varargin)

%Variables utilizadas por la funcion
global DATOS;

%Crea la figura y la oculta mientras la configura
DATOS.h(2)=figure('visible','off');
%Configuracion de la figura
```

```
set(DATOS.h(2),'position',[100 140 600 300],'color',[0 0 0]);
set(DATOS.h(2),'menubar','none','name','Acerca de
SenSING','numbertitle','off','windowstyle','modal');
set(DATOS.h(2),'resize','off','visible','on','deletefcn',['global
DATOS;','delete(DATOS.h(2));'],'keypressfcn',['global DATOS;','delete(DATOS.h(2));'], 'Units',
'pixels');
%Creacion de las ejes para la imagen
DATOS.ejes(8)=axes;
%Carga la imagen
imagen = imread('Acerca_de', 'png');
%Muestra la imagen
image(imagen);
%Reconfiguracion de los ejes
set(DATOS.ejes(8),'Visible', 'off','Units', 'pixels','Position', [0 0 600 300 ]);
%---------------------------------------------------------------------
%FUNCIONES INTERNAS
%---------------------------------------------------------------------
function [varargout]=ActualizarCanal(varargin)

%Definicion de variables
global DATOS;

%Numero del canal
set(DATOS.numcanal,'string',strcat('Canal_',num2str(DATOS.Actual)));

%Nombre de la variable medida
set(DATOS.nomvar,'string',DATOS.vc{DATOS.Actual,1});

%Valores minimo y maximo de la variable
set(DATOS.varmin,'string',DATOS.ValMinC(DATOS.Actual));
set(DATOS.varmax,'string',DATOS.ValMaxC(DATOS.Actual));

%Valores minimo y maximo para la visualizacion
set(DATOS.vismin,'string',DATOS.VisMinC(DATOS.Actual));
set(DATOS.vismax,'string',DATOS.VisMaxC(DATOS.Actual));
%---------------------------------------------------------------------
%---------------------------------------------------------------------
function [varargout]=Habilitar(varargin)

%Definicion de variables
global DATOS;
```

```
%Abre el puerto
fopen(DATOS.Puerto);
%Transmite el comando
fprintf(DATOS.Puerto,'%s',num2str(DATOS.Actual),'async');
%Detiene el programa
pause(0.2);
%Cierra el puerto
fclose(DATOS.Puerto);
%Obtiene los datos
DATOS.vc{DATOS.Actual,1}=get(DATOS.nomvar,'string');
DATOS.ValMinC(DATOS.Actual)=str2num(get(DATOS.varmin,'string'));
DATOS.ValMaxC(DATOS.Actual)=str2num(get(DATOS.varmax,'string'));
DATOS.VisMinC(DATOS.Actual)=str2num(get(DATOS.vismin,'string'));
DATOS.VisMaxC(DATOS.Actual)=str2num(get(DATOS.vismax,'string'));
%Actualiza los controles
set(DATOS.TeCanal(DATOS.Actual),'string',DATOS.vc{DATOS.Actual,1});
set(DATOS.TeLectura(DATOS.Actual),'string','0');
%set(DATOS.EsGrafica(DATOS.Actual),'enable','on');
DATOS.Bandera(DATOS.Actual)=1;

%Prepara los ejes
axes(DATOS.ejesg(DATOS.Actual));
DATOS.grafica(DATOS.Actual)=plot(0,0);
set(DATOS.grafica(DATOS.Actual),'color',[1 1 0]);
axis([0 62 DATOS.VisMinC(DATOS.Actual) DATOS.VisMaxC(DATOS.Actual)]);
set(DATOS.ejesg(DATOS.Actual),'visible','off','drawmode','fast');

%Inicializa el contador de lecturas
DATOS.contador(DATOS.Actual)=0;

%Calcula y almacena las escalas
DATOS.Factor(DATOS.Actual)=(DATOS.ValMaxC(DATOS.Actual)-
DATOS.ValMinC(DATOS.Actual))/511;

if DATOS.ConfigurarCorriente==0
   if DATOS.Actual<3
      %Habilita el boton siguiente
      set(DATOS.siguiente,'enable','on');
   else
      delete(DATOS.cv);
```

```
        %Deshabilita el menu Voltaje
        set(DATOS.MeVoltaje,'enable','off');
    end
else
    if DATOS.Actual<6
        %Habilita el boton siguiente
        set(DATOS.siguiente,'enable','on');
    else
        delete(DATOS.cv);
        %Deshabilita el menu Voltaje
        set(DATOS.MeCorriente,'enable','off');
    end
end

%Numero de canales habilitados
DATOS.Cantidad=DATOS.Cantidad+1;
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
function [varargout]=Defecto(varargin)

%Definicion de variables
global DATOS;

set(DATOS.MeCanales,'enable','off');
set(DATOS.MeIniciar,'enable','on');

%Actualiza los controles
for k=1:6
    set(DATOS.TeCanal(k),'string',DATOS.vc{k,1});
    set(DATOS.TeLectura(k),'string','0');
    %set(DATOS.EsGrafica(k),'enable','on');
    DATOS.Bandera(k)=1;

    %Prepara los ejes
    axes(DATOS.ejesg(k));
    DATOS.grafica(k)=plot(0,0);
    set(DATOS.grafica(k),'color',[1 1 0]);
    axis([0 62 DATOS.VisMinC(k) DATOS.VisMaxC(k)]);
    set(DATOS.ejesg(k),'visible','off','drawmode','fast');
```

```
    %Inicializa el contador de lecturas
    DATOS.contador(k)=0;
end

%Abre el puerto
fopen(DATOS.Puerto);
%Transmite el comando
fprintf(DATOS.Puerto,'%s','7','async');
%Detiene la ejecucion del programa
pause(0.2);
%Cierra el puerto
fclose(DATOS.Puerto);

%Calcula y almacena las escalas
%72.5
for k=1:3
    calibravol=1;
    DATOS.Factor(k)=((DATOS.ValMaxC(k)-DATOS.ValMinC(k))/1023)*calibravol;
end
for k=4
    calibracorr=1.35;
    DATOS.Factor(k)=((DATOS.ValMaxC(k)-DATOS.ValMinC(k))/1023)*calibracorr;
end
for k=5
    calibracorr1=1.64;
    DATOS.Factor(k)=((DATOS.ValMaxC(k)-DATOS.ValMinC(k))/1023)*calibracorr1;
end
for k=6
    calibracorr2=1.7;
    DATOS.Factor(k)=((DATOS.ValMaxC(k)-DATOS.ValMinC(k))/1023)*calibracorr2;
end
%Numero de canales habilitados
DATOS.Cantidad=6;
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
function [varargout]=ReConfigurarPuerto(varargin)

%Definicion de variables
```

```
global DATOS;

%Obtiene y almacena la nueva configuracion
DATOS.Configuracion(1)=get(DATOS.CoBPOR,'value');
DATOS.Configuracion(2)=get(DATOS.CoBPS,'value');
DATOS.Configuracion(3)=get(DATOS.CoBDAT,'value');
DATOS.Configuracion(4)=get(DATOS.CoBP,'value');
DATOS.Configuracion(5)=get(DATOS.CoBS,'value');
DATOS.Configuracion(6)=get(DATOS.CoFLU,'value');

%Configura el puerto

switch DATOS.Configuracion(1)
  case 1
    set(DATOS.Puerto,'port','COM1');
  case 2
    set(DATOS.Puerto,'port','COM2');
  case 3
    set(DATOS.Puerto,'port','COM3');
  case 4
    set(DATOS.Puerto,'port','COM4');
  case 5
    set(DATOS.Puerto,'port','COM5');
  case 6
    set(DATOS.Puerto,'port','COM6');
end

switch DATOS.Configuracion(2)
  case 1
    set(DATOS.Puerto,'baudrate',300);
  case 2
    set(DATOS.Puerto,'baudrate',1200);
  case 3
    set(DATOS.Puerto,'baudrate',2400);
  case 4
    set(DATOS.Puerto,'baudrate',4800);
  case 5
    set(DATOS.Puerto,'baudrate',9600);
  case 6
```

```
      set(DATOS.Puerto,'baudrate',19200);
   case 7
      set(DATOS.Puerto,'baudrate',38400);
   case 8
      set(DATOS.Puerto,'baudrate',57600);
   case 9
      set(DATOS.Puerto,'baudrate',115200);
end

switch DATOS.Configuracion(3)
   case 1
      set(DATOS.Puerto,'databits',5);
   case 2
      set(DATOS.Puerto,'databits',6);
   case 3
      set(DATOS.Puerto,'databits',7);
   case 4
      set(DATOS.Puerto,'databits',8);
end

switch DATOS.Configuracion(4)
   case 1
      set(DATOS.Puerto,'parity','even');
   case 2
      set(DATOS.Puerto,'parity','odd');
   case 3
      set(DATOS.Puerto,'parity','none');
   case 4
      set(DATOS.Puerto,'parity','mark');
   case 5
      set(DATOS.Puerto,'parity','space');
end

switch DATOS.Configuracion(5)
   case 1
      set(DATOS.Puerto,'stopbits',1);
   case 2
      set(DATOS.Puerto,'stopbits',2);
end
```

```
switch DATOS.Configuracion(6)
   case 1
      set(DATOS.Puerto,'flowcontrol','software');
   case 2
      set(DATOS.Puerto,'flowcontrol','hardware');
   case 3
      set(DATOS.Puerto,'flowcontrol','none');
end

delete(DATOS.Configurar);
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
function [varargout]=ReConfigurarBluePort1(varargin)

%Definicion de variables
global DATOS;

%Obtiene y almacena la nueva configuracion
for k=1:5
   nueva(k)=get(DATOS.BlueConf(k),'value');
end

switch nueva(1)
   case 1
      velocidad='19200';
   case 2
      velocidad='38400';
   case 3
      velocidad='57600';
   case 4
      velocidad='115200';
end

switch nueva(2)
   case 1
      bd='7';
   case 2
      bd='8';
```

```
end

switch nueva(3)
   case 1
      par='E';
   case 2
      par='O';
   case 3
      par='N';
end

switch nueva(4)
   case 1
      bs='1';
   case 2
      bs='2';
end

switch nueva(5)
   case 1
      flow=',CTS';
   case 2
      flow='';
end

 %Conforma la instruccion
Instruccion=strcat('S0',velocidad,',',bd,par,bs,flow);
 %Abre el puerto
fopen(DATOS.Puerto);
 %Envia la instruccion
fprintf(DATOS.Puerto,Instruccion);
 %Inicia el temporizador
DATOS.t1 = timer('TimerFcn','fclose(DATOS.Puerto)', 'StartDelay',1);
start(DATOS.t1);
 %--------------------------------------------------------------------
 %--------------------------------------------------------------------
function [varargout]=ReConfigurarBluePort2(varargin)

 %Definicion de variables
```

```
global DATOS;

if DATOS.set==0
   DATOS.Leido=DATOS.Leido+1;
   switch DATOS.Leido
     case 1
        Instruccion=strcat('?F0');
     case 2
        Instruccion=strcat('?S2');
     case 3
        Instruccion=strcat('?S4');
     case 4
        Instruccion=strcat('?L');
   end
else
   switch DATOS.set
      case 1
         Instruccion=strcat('F0',get(DATOS.BlueConf(6),'string'));
      case 2
         Instruccion=strcat('S2',get(DATOS.BlueConf(7),'string'));
      case 3
         Instruccion=strcat('S4',get(DATOS.BlueConf(8),'string'));
      case 4
         opcion=get(DATOS.BlueConf(10),'value');
         switch opcion
            case 1
               parametro='0';
            case 2
               parametro='1';
            case 3
               parametro='2';
            case 4
               parametro='3';
         end
         Instruccion=strcat('L',parametro);
   end
end

%Abre el puerto
```

```
fopen(DATOS.Puerto);
%Configura el nombre del dispositivo
fprintf(DATOS.Puerto,Instruccion);
%Inicia el temporizador
DATOS.t1 = timer('TimerFcn','fclose(DATOS.Puerto)', 'StartDelay',1);
start(DATOS.t1);
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
function [varargout]=Recibir(varargin)

%Definicion de variables
global DATOS;

if DATOS.DetenerA==1
   fprintf(DATOS.Puerto,'%s','DD','async');
   stop(DATOS.t2);
end

Numero=DATOS.Puerto.bytesavailable;

if DATOS.AsistenteEnab==1
   switch DATOS.paso
      case 1
         %Detiene el temporizador
         stop(DATOS.t1);
         %Lee los dos datos correspondientes a la instruccion
         if get(DATOS.Puerto,'bytesavailable')>=2
            fread(DATOS.Puerto,2);
         end
         %Lee la parte restante de la respuesta
         r=strvcat(char(fread(DATOS.Puerto,get(DATOS.Puerto,'bytesavailable')-1)))';
         fread(DATOS.Puerto,1);
         set(DATOS.PasosAsistente(3),'string',r);
         %Cierra el puerto
         fclose(DATOS.Puerto);
         %Habilita el control para continuar con el Asistente
         set(DATOS.siguienteAsist,'enable','on');

      case 2
```

```
        %Detiene el temporizador
        stop(DATOS.t1);
        %Espera ACK de la nueva configuracion
        r=strvcat(char(fread(DATOS.Puerto,get(DATOS.Puerto,'bytesavailable')-1)))';
        fread(DATOS.Puerto,1);
        %Cierra el puerto
        fclose(DATOS.Puerto);
        %Si la respuesta es un ACK, entonces reconfigura el puerto
        %serial y cierra el asistente
        if isequal(r,'ACK')
           %Presenta la advertencia
           h=warndlg('Esta acción cerrará el Asistente Bluetooth. Si modificó algun parámetro,
recuerde reconfigurar el puerto antes de abrirlo nuevamente.','SenSING V1.0');
           %Espera el reconocimiento del mensaje
           waitfor(h);
           delete(DATOS.h(3));
           DATOS.paso=0;
           DATOS.AsistenteEnab=0;
        end

     case 3
        auxiliar1
     case 4
        if DATOS.set<=3
           %Detiene el temporizador
           stop(DATOS.t1);
           %Espera ACK de la nueva configuracion
           r=strvcat(char(fread(DATOS.Puerto,get(DATOS.Puerto,'bytesavailable')-1)))';
           fread(DATOS.Puerto,1);
           %Cierra el puerto
           fclose(DATOS.Puerto);
           %Si la respuesta es un ACK, entonces reconfigura el puerto serial y cierra el asistente
           if isequal(r,'ACK')
              DATOS.set=DATOS.set+1;
              ReConfigurarBluePort2
           end
        else
           delete(DATOS.h(3));
           DATOS.paso=0;
           DATOS.AsistenteEnab=0;
```

```
        end
      end
else

   if Numero==DATOS.Cantidad*9+1
      for k=1:DATOS.Cantidad
         %Lectura del canal
         canal=str2num(char(fread(DATOS.Puerto,1)));
         %_____________________________________
        % for canal=1:6

         %Lectura del dato

dato=str2num(char(fread(DATOS.Puerto,1)))*1000+str2num(char(fread(DATOS.Puerto,1)))*10
0+...
           str2num(char(fread(DATOS.Puerto,1)))*10+str2num(char(fread(DATOS.Puerto,1)));
         %Lectura del complemento

complemento=str2num(char(fread(DATOS.Puerto,1)))*1000+str2num(char(fread(DATOS.Puerto
,1)))*100+...
           str2num(char(fread(DATOS.Puerto,1)))*10+str2num(char(fread(DATOS.Puerto,1)));


        if dato+complemento==9999
          %Actualiza los ejes para la grafica
          axes(DATOS.ejesg(canal));
          %Obtiene los datos actuales de la grafica
          vx=get(DATOS.grafica(canal),'xdata');
          vy=get(DATOS.grafica(canal),'ydata');
          %Actualiza el contador de datos


          DATOS.contador(canal)=DATOS.contador(canal)+1;
          %Actualiza la matriz de datos
          %X=dato*DATOS.Factor(canal);
          %aprox=X-1e-10*X^6+6e-8*X^5-1e-5*X^4+8e-4*X^3-3.39e-
2*X^2+0.6124*X+2.7214;
          %DATOS.matriz(DATOS.contador(canal),canal)=aprox;
          %for canal=1:3
            if dato<138
             DATOS.matriz(DATOS.contador(canal),canal)=0;


            else
```

```
            DATOS.matriz(DATOS.contador(canal),canal)=dato*DATOS.Factor(canal)*0.88;
           end
         % end
          %for canal=4:6
             %DATOS.matriz(DATOS.contador(canal),canal)=dato*DATOS.Factor(canal);
          %end
          %Actualiza la grafica
          if DATOS.contador(canal)<=60
           vx(DATOS.contador(canal))=DATOS.contador(canal);
           vy(DATOS.contador(canal))=DATOS.matriz(DATOS.contador(canal),canal);
           set(DATOS.grafica(canal),'xdata',vx,'ydata',vy);
          else
           vy(61)=DATOS.matriz(DATOS.contador(canal),canal);
           set(DATOS.grafica(canal),'xdata',vx,'ydata',vy(2:61));
          end
          %Actualiza el scroll
          set(DATOS.EsGrafica(canal),'value',DATOS.contador(canal));
          %Actualiza el cuadro de texto
          y=num2str(DATOS.matriz(DATOS.contador(canal),canal),'%3.2f');
          set(DATOS.TeLectura(canal),'string',y);
          %Actualiza la hoja de Excel
          eActiveSheetRange = Range(DATOS.eActiveSheet,
strcat(char(64+canal),num2str(DATOS.contador(canal))));
          set(eActiveSheetRange, 'Value', str2num(y));
          end
         %end
     %_____________________________________________________

     end %el end del for
        %Lectura del terminator
        str2num(char(fread(DATOS.Puerto,1)));
   else
     if Numero~=0
        fread(DATOS.Puerto,Numero);
     end
   end
end

if DATOS.DetenerA==1
```

```
    DATOS.DetenerA=0;
    set(DATOS.MeDetener,'enable','off');
    pause(0.2);
    fclose(DATOS.Puerto);

    Quit(DATOS.e);
    delete(DATOS.e);

    %Habilita los scroll
    for k=1:6
        if DATOS.Bandera(k)==1
            set(DATOS.EsGrafica(k),'enable','on','callback',@scroll);
        end
    end
    %Guarda los valores iniciales
    DATOS.InicioScroll=get(DATOS.EsGrafica,'value');
    DATOS.InicioScroll=cell2mat(DATOS.InicioScroll);
end
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
function [varargout]=auxiliar1(varargin)

global DATOS;

if DATOS.Leido<=3
    %Detiene el temporizador
    stop(DATOS.t1);
    %Lee los dos datos correspondientes a la instruccion
    if get(DATOS.Puerto,'bytesavailable')>=2
        fread(DATOS.Puerto,2);
    end
    %Lee la parte restante de la respuesta
    r=strvcat(char(fread(DATOS.Puerto,get(DATOS.Puerto,'bytesavailable')-1)))';
    fread(DATOS.Puerto,1);
    set(DATOS.BlueConf(DATOS.Leido+5),'string',r);
    %Cierra el puerto
    fclose(DATOS.Puerto);
    %Llama la funcion de lectura
    ReConfigurarBluePort2
```

```
else
    %Detiene el temporizador
    stop(DATOS.t1);
    %Lee los dos datos correspondientes a la instruccion
    if get(DATOS.Puerto,'bytesavailable')>=1
        fread(DATOS.Puerto,1);
    end
    %Lee la parte restante de la respuesta
    r=strvcat(char(fread(DATOS.Puerto,get(DATOS.Puerto,'bytesavailable')-1)))';
    fread(DATOS.Puerto,1);
    switch r
        case '0'
            set(DATOS.BlueConf(10),'value',1);
        case '1'
            set(DATOS.BlueConf(10),'value',2);
        case '2'
            set(DATOS.BlueConf(10),'value',3);
        case '3'
            set(DATOS.BlueConf(10),'value',4);
    end
    %Cierra el puerto
    fclose(DATOS.Puerto);
end
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
%Controla redimensionamiento de la pantalla
function [varargout]=maximizar(varargin)

global DATOS;

if DATOS.estadopantalla==0;
     set(DATOS.ejes(7),'position',[0 35 1250 850]);

    set(DATOS.ejesg(1),'position',[75 690 223 145]);
    set(DATOS.ejesg(2),'position',[75 400 223 145]);
    set(DATOS.ejesg(3),'position',[75 150 223 145]);
    set(DATOS.ejesg(4),'position',[953 680 223 122]);
    set(DATOS.ejesg(5),'position',[953 420 223 122]);
    set(DATOS.ejesg(6),'position',[953 170 223 122]);
```

```
    set(DATOS.TeLectura(1),'position',[75 630 104 20 ]);
    set(DATOS.TeLectura(2),'position',[75 375 104 20 ]);
    set(DATOS.TeLectura(3),'position',[75 120 104 20 ]);
    set(DATOS.TeLectura(4),'position',[1075 630 104 20]);
    set(DATOS.TeLectura(5),'position',[1075 375 104 20]);
    set(DATOS.TeLectura(6),'position',[1075 120 104 20 ]);

    set(DATOS.EsGrafica(1),'position',[75 669 230 18]);
    set(DATOS.EsGrafica(2),'position',[75 413 230 18]);
    set(DATOS.EsGrafica(3),'position',[75 157 230 18]);
    set(DATOS.EsGrafica(4),'position',[953 669 230 18]);
    set(DATOS.EsGrafica(5),'position',[953 413 230 18]);
    set(DATOS.EsGrafica(6),'position',[953 157 230 18]);

    set(DATOS.TeCanal(1),'position',[75 645 104 20 ]);
    set(DATOS.TeCanal(2),'position',[75 390 104 20 ]);
    set(DATOS.TeCanal(3),'position',[75 135 104 20 ]);
    set(DATOS.TeCanal(4),'position',[1075 645 104 20 ]);
    set(DATOS.TeCanal(5),'position',[1075 390 104 20 ]);
    set(DATOS.TeCanal(6),'position',[1075 135 104 20 ]);

    DATOS.estadopantalla=1;
else
     set(DATOS.ejes(7),'position',[0 35 1250 850]);

    set(DATOS.ejesg(1),'position',[75 690 223 145]);
    set(DATOS.ejesg(2),'position',[75 400 223 145]);
    set(DATOS.ejesg(3),'position',[75 150 223 145]);
    set(DATOS.ejesg(4),'position',[953 680 223 122]);
    set(DATOS.ejesg(5),'position',[953 420 223 122]);
    set(DATOS.ejesg(6),'position',[953 170 223 122]);

    set(DATOS.TeLectura(1),'position',[75 630 104 20 ]);
    set(DATOS.TeLectura(2),'position',[75 375 104 20 ]);
    set(DATOS.TeLectura(3),'position',[75 120 104 20 ]);
    set(DATOS.TeLectura(4),'position',[1075 630 104 20]);
    set(DATOS.TeLectura(5),'position',[1075 375 104 20]);
    set(DATOS.TeLectura(6),'position',[1075 120 104 20 ]);
```

```
    set(DATOS.EsGrafica(1),'position',[75 669 230 18]);
    set(DATOS.EsGrafica(2),'position',[75 413 230 18]);
    set(DATOS.EsGrafica(3),'position',[75 157 230 18]);
    set(DATOS.EsGrafica(4),'position',[953 669 230 18]);
    set(DATOS.EsGrafica(5),'position',[953 413 230 18]);
    set(DATOS.EsGrafica(6),'position',[953 157 230 18]);

    set(DATOS.TeCanal(1),'position',[75 645 104 20 ]);
    set(DATOS.TeCanal(2),'position',[75 390 104 20 ]);
    set(DATOS.TeCanal(3),'position',[75 135 104 20 ]);
    set(DATOS.TeCanal(4),'position',[1075 645 104 20 ]);
    set(DATOS.TeCanal(5),'position',[1075 390 104 20 ]);
    set(DATOS.TeCanal(6),'position',[1075 135 104 20 ]);

    DATOS.estadopantalla=0;
end
%--------------------------------------------------------------------------
```