UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Diseño e implementación de una red Bluetooth de comunicación entre 5 sensores de signos vitales y un teléfono celular

Trabajo de Graduación presentado por
Juana Marlenne Rivera Marroquín para optar al grado académico de
Licenciada en Ingeniería Electrónica

Guatemala
2012

Diseño e implementación de una red Bluetooth de comunicación entre 5 sensores de signos vitales y un teléfono celular

UNIVERSIDAD DEL VALLE DE GUATEMALA

Facultad de Ingeniería

Diseño e implementación de una red Bluetooth de comunicación entre 5 sensores de signos vitales y un teléfono celular

Trabajo de Graduación presentado por

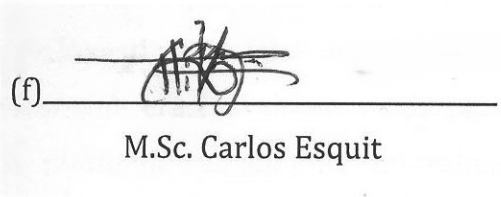
Juana Marlenne Rivera Marroquín para optar al grado académico de

Licenciada en Ingeniería Electrónica

Guatemala

2012

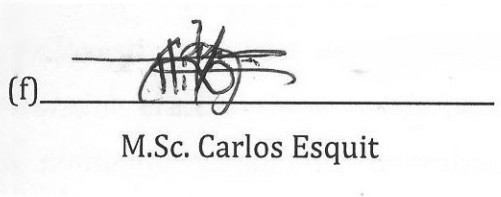
Vo.Bo.:



(f)_____

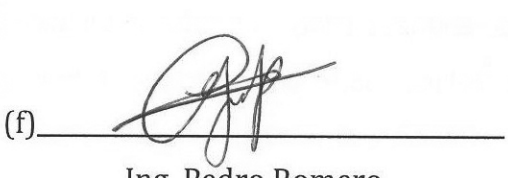
M.Sc. Carlos Esquit

Tribunal:



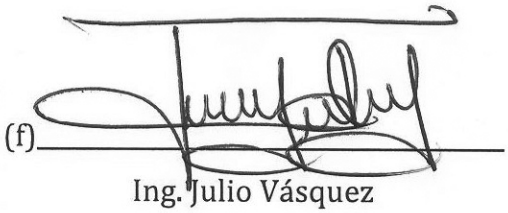
(f)_____

M.Sc. Carlos Esquit



(f)_____

Ing. Pedro Romero



(f)_____

Ing. Julio Vásquez

Fecha de Aprobación: Guatemala, 13 de junio de 2012

PREFACIO

Este proyecto nació del deseo de viajar a Europa a formar parte de un proyecto en conjunto con la Universidad Politécnica de Madrid. Luego de luchar para lograr el patrocinio, no se logró llevar a cabo el proyecto en conjunto. De esta cuenta, se platicó con un grupo y decidimos continuar por nuestra cuenta con un proyecto innovador abriendo una nueva línea de investigación en el departamento.

El área de la telemedicina era casi desconocida para mí, y no tenía idea del mundo en el que me estaba metiendo cuando inicié mi trabajo en el proyecto. El poder asistir a personas sin estar físicamente presentes me parece uno de los mayores logros de la electrónica y sin duda alguna quisiera continuar incursionando en ella.

A pesar de haber iniciado este proyecto de manera incierta, con problemas de ajuste, despegamos con alas firmes y llegamos a la meta. A lo largo del trayecto trabajé cercanamente al Megaproyecto Konócete, un fabuloso grupo, con quienes me uní bastante, y no me queda más que agradecer a Luis Juárez, Isaac Mejía y Clara Cruz, los integrantes, por haber luchado a mi lado para completar esta proeza. También merecen mi agradecimiento tres personas: la Ing. Marie André Destarac, quien debió retirarse antes de finalizar el proyecto, no sin antes asistirme con sus consejos, el Ing. Luis Fernando Reina, por su disponibilidad para resolver cualquier tipo de duda y hacer hasta lo imposible por ayudarnos y a M.Sc. Carlos Esquit por su constante presencia y apoyo como director del departamento y asesor.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

RESUMEN

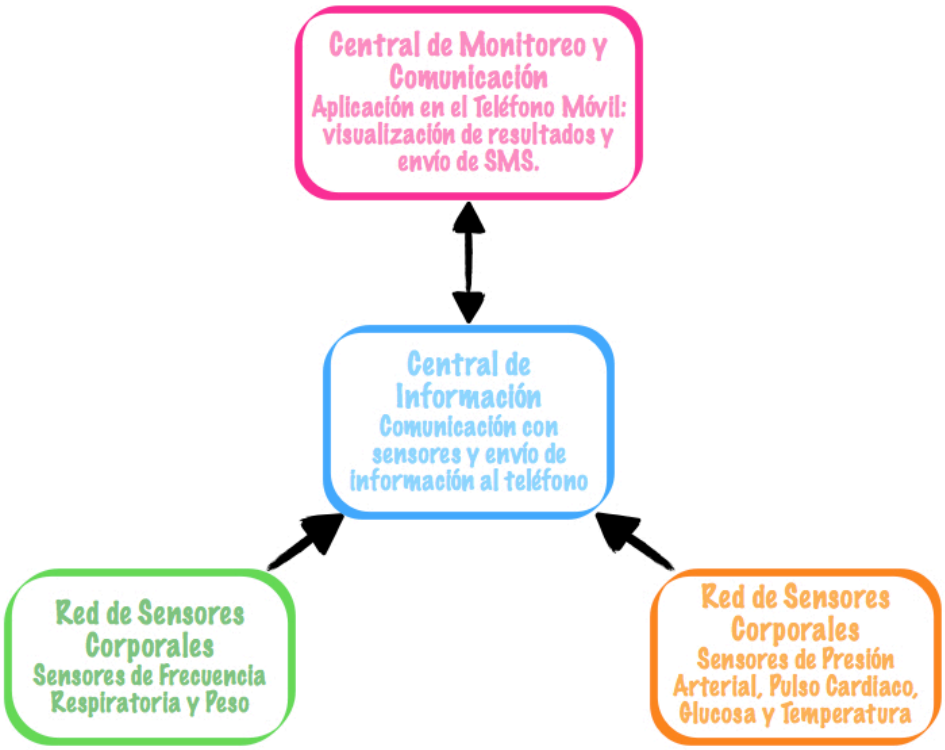
Este trabajo de graduación es parte de una red de monitoreo de signos vitales. El proyecto general comprende el diseño e implementación de una red de monitoreo de sensores de signos vitales con comunicación a un teléfono celular para informar acerca del estado de salud del paciente. De esta manera el proyecto busca unificar en un solo sistema las mediciones correspondientes a un típico examen de signos vitales, con lo cual se espera reducir el tiempo en la toma de esta información, manteniendo un mejor control de la salud del paciente y mejorando la productividad y eficacia del hospital o doctor que lo utilice.

Este trabajo de graduación tratará del diseño e implementación de la central de información. La red de comunicación contará con una interfaz de comunicación por medio Bluetooth para establecer contacto entre la unidad de control y el teléfono celular del paciente. Adicionalmente, esta red se conectará utilizando la misma tecnología con cada uno de los sensores. De esta manera, la central de información realizará las conexiones y obtendrá la información de los sensores. Una vez obtenida esta información esta central ordenará los datos en un formato compatible con el teléfono celular y se los enviará para continuar con el análisis de la información.

I. INTRODUCCIÓN

Este proyecto trata acerca del desarrollo de un dispositivo de análisis y observación de los signos vitales en personas con discapacidades o mayores a 18 años que necesiten un monitoreo frecuente. Dicho dispositivo es adaptable al usuario para que lo utilicen hombres y mujeres, incluyendo adultos mayores. El mismo cuenta con una unidad de control inteligente para recibir, gestionar y procesar información vital del individuo al que asiste, y de acuerdo con los resultados obtenidos envía mensajes de texto al cuidador o médico del paciente. El diagrama de bloques de la Figura 1 muestra los módulos en los que se divide el proyecto.

Figura 1: Diagrama de bloques de la red sensorial de signos vitales.



La central de información, de cuyo desarrollo trata este Trabajo de Graduación, es la encargada de reunir los datos provenientes de una red de sensores que el usuario debe utilizar. Entre las mediciones a realizar, tal como muestra el diagrama, se encuentran la presión arterial, pulso cardíaco, nivel de glucosa, frecuencia

respiratoria, peso y temperatura. Estos sensores envían la información a la unidad de control que es la encargada de ordenar esta información para facilitar su análisis por el teléfono celular. De esta forma el propio usuario, sus cuidadores y/o médicos obtienen un monitoreo constante del estado general del usuario.

El sistema cuenta con una interfaz de comunicación por medio Bluetooth para establecer contacto entre la unidad de control y el teléfono celular del paciente. A su vez, el teléfono celular envía mensajes de texto con la información de los signos vitales medidos. De esta manera, el dispositivo permite al cuidador estar informado del estado de salud general del paciente frecuentemente de una manera más práctica. Adicionalmente, si durante una medición algún parámetro excede sus niveles normales, el sistema inteligente envía mensajes de alerta mostrando cuál medición muestra el problema.

Las partes cruciales del proyecto se encuentran en la comunicación con cada sensor para garantizar la integridad de las mediciones y en el óptimo funcionamiento de la central de información. Durante el proceso de implementación se cuidaron estos dos detalles para presentar un producto final funcionando adecuadamente. Con respecto a la comunicación se utilizó el Bluetooth, tecnología inalámbrica que fue diseñada para la implementación de redes con pocos dispositivos, características óptimas para la red sensorial propuesta. Adicionalmente, se integró una pantalla para mostrar mensajes explicativos al usuario a manera de mantenerlo al tanto con las acciones que se encuentre ejecutando la central de información. Integrando estas partes al sistema se logró la implementación exitosa de la obtención de información de los sensores y la retransmisión de esta información al teléfono celular.

II. OBJETIVOS

A. OBJETIVO GENERAL

Diseñar e implementar una central de información que obtenga información de la red de sensores, la coloque en un formato entendible por el teléfono celular y se la envíe al móvil.

B. OBJETIVOS ESPECÍFICOS

1. Diseñar y construir un sistema de comunicación Bluetooth entre la central de información y el módulo de monitoreo y comunicación.

2. Diseñar y construir un sistema de comunicación Bluetooth entre la central de información y cada uno de los sensores implementados.

3. Diseñar e implementar una central de información para los signos vitales medidos que tenga una interfaz para mostrar el nivel de las variables según se van recibiendo.

III. MARCO TEÓRICO

A. TECNOLOGÍA BLUETOOTH

La tecnología Bluetooth fue creada por la compañía Bluetooth Special Interest Group (SIG) quienes se encargan de administrar todo lo relacionado con esta tecnología como lo es el desarrollo y la actualización del mismo. Esta compañía no fabrica los productos que emplean esta tecnología sino que se encargan de publicar las especificaciones de la tecnología, administrar el programa de acreditación y proteger las patentes de la marca para que esté disponible a todos. A continuación se muestran las características más importantes de esta tecnología que fueron empleadas en el desarrollo de este trabajo (Bluetooth SIG, 2010).

1. Canal de radio. La capa física del Bluetooth opera en la banda de frecuencias ISM (Industrial, Científica y Médica por sus siglas en inglés) a 2.4GHz. El sistema emplea un transmisor receptor que salta de frecuencias para prevenir interferencia y que se atenúe la señal. La operación de radio frecuencia utiliza una modulación de frecuencias binaria para minimizar la complejidad del transmisor receptor y tiene un bit rate de 1 Mbps, aunque también existe el modo de envío de datos mejorado (Enhanced Data Rate) que pueden transmitir a 2 ó 3 Mbps. Cuando se utiliza una conexión Bluetooth típica, entre los dispositivos se comparten un sólo canal de radio físico que está sincronizado a un reloj común con el mismo patrón de salto de frecuencias (Bluetooth SIG, 2010).

2. Redes Bluetooth piconet y saltos de frecuencia. Una piconet es una red ad-hoc que une a un grupo de dispositivos usando el canal físico del Bluetooth y consiste en un maestro y uno o varios esclavos. El maestro es el dispositivo que proporciona el reloj para que los esclavos se sincronicen y además proporciona el patrón del cambio de frecuencias. También es importante mencionar que el dispositivo maestro nunca solicita la conexión, este solo espera a que dispositivos se quieran conectar a él, y es el esclavo quien busca los dispositivos cercanos y se conecta con el maestro (Bluetooth SIG, 2010).

Una vez se haya establecido la conexión, los dispositivos utilizarán un patrón de salto de frecuencias que se calcula utilizando un algoritmo que utiliza la dirección Bluetooth del dispositivo y el reloj del maestro. El salto de frecuencias terminan siendo 79 frecuencias escogidas en un proceso cuasi-aleatorio en el rango de la banda ISM. Este mismo patrón puede ser modificado por el dispositivo para excluir frecuencias que estén causando interferencia debido a otros dispositivos que utilizan esta banda pero no utilizan el algoritmo de salto de frecuencia (Bluetooth SIG, 2010).

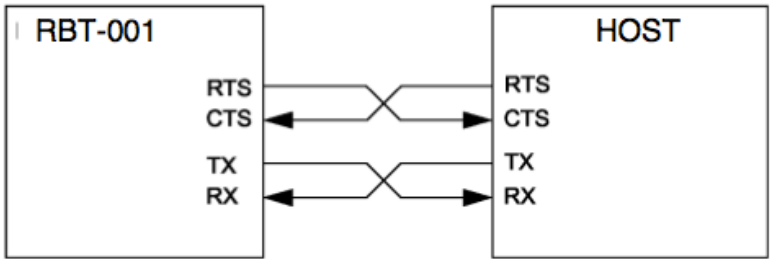
3. Transmisión de datos full-dúplex. Durante la conexión entre dos o más dispositivos el canal físico se divide en unidades de tiempo o espacios para la transmisión de datos. Los datos se transmiten en paquetes que se envían dentro de estos espacios de tiempo. Cuando los paquetes de datos son más grandes y si las circunstancias lo permiten, un número de estos espacios de tiempo pueden ser asignados de manera contigua para que formen un solo paquete. Es importante notar que el salto de frecuencias ocurre únicamente entre la transmisión y recepción de datos, nunca ocurre durante una transmisión. Además, la tecnología Bluetooth utiliza un esquema de división de tiempo para tener una comunicación full dúplex (Bluetooth SIG, 2010).

B. MÓDULO ROBOTECH BLUETOOTH SERIAL (RBT-001)

Además de ser necesario comprender las especificaciones de la tecnología Bluetooth, fue muy importante estudiar el funcionamiento del dispositivo RBT-001 que es el que se utilizará en el Sistema de Monitoreo para comunicar el teléfono celular con el microcontrolador que contendrá la información de los sensores. Este dispositivo es compatible con las especificaciones 1.x y 2.0 de Bluetooth y es un dispositivo de bajo consumo de potencia. Al cumplir con las especificaciones de Bluetooth, el dispositivo soporta AFH (Salto de Frecuencias Adaptables por sus siglas en inglés) y los perfiles SDAP y SPP de Bluetooth. El perfil SDAP - Service Discovery Aplication Profile describe cómo un dispositivo debe buscar los servicios de otro dispositivo Bluetooth para determinar de qué manera pueden comunicarse. El perfil

SPP define cómo configurar puertos seriales para la transferencia de datos (Parallax, Inc., 2010)

Figura 2: Configuración serial null módem.



(Parallax, Inc., 2010)

El Robotech RBT-001 también tiene una interfaz RS-232 para conectarse con otros dispositivos. El baud rate máximo al que este dispositivo puede enviar y transmitir datos es de 921.6k y tiene la ventaja que cuenta con una comunicación serial full-dúplex, teniendo así un canal para enviar y uno para recibir información. Además, el dispositivo se puede conectar utilizando la configuración Null-Modem para utilizar los bits RTS y DTS del RS-232 como bits de control del flujo de datos tal como se muestra en la Figura 1. La configuración null módem funciona a nivel físico, pero el RBT-001 también tiene un protocolo que se debe seguir cuando se desea configurar las distintas funciones del dispositivo tal como se muestra en la Figura 3. Este framing muestra que el RBT-001 tiene un set de instrucciones y opcodes para cada perfil Bluetooth que maneja. Cada uno de los bloques mostrados en la Figura 3, tomada del manual de usuario del RBT-001 (Parallax, Inc., 2010), sirve para identificar qué función realiza cada parte del paquete completo:

a. Start Delimiter: indica que los siguientes datos enviados no es información para el dispositivo sino datos para ese chip.
b. Tipo del paquete indica si es una petición (request), confirmación, indicación o respuesta.
c. Opcode: especifica lo que hará la instrucción.
d. Longitud de los datos a enviar.
e. Checksum: suma de los valores del tipo de paquete, opcode y longitud de datos para verificar que se recibieron correctamente los paquetes.

f. Datos: envío de los datos con la longitud especificada.

g. End Delimiter: fin del comando.

Figura 3: Framing para los comandos del dispositivo RBT-001.

Start delimiter	Packet Type identification	Op code	Data length	Check-sum	Packet Data	End delimiter
1 byte	1 byte	1 byte	2 bytes	1 byte	<Data length> bytes	1 byte
	I---------- Checksum ----------I					

(Parallax, Inc., 2010)

IV. ANTECEDENTES

La telemetría ha formado una rama completa de investigación alrededor del mundo. Varios países están trabajando en nuevos proyectos a manera de obtener avances útiles para la comunidad. En Guatemala, se puede observar que esta rama de la medicina e ingeniería es una solución para facilitar la atención médica en distintos puntos del país. El cuidado del paciente en casa, la atención de pacientes en sitios remotos de la ciudad o incluso la implementación de este tipo de tecnologías en hospitales públicos pretende ser una herramienta útil para mejorar la calidad de vida de las personas.

La idea de este proyecto nació originalmente de un trabajo en conjunto entre la Universidad del Valle de Guatemala y la Universidad Politécnica de Madrid. El proyecto original proponía diseñar e implementar un bastón asistencial con sensores de signos vitales para monitorear el estado de salud del paciente y asistirlo a caminar. Luego de problemas para conseguir patrocinio, entre otros, se decidió únicamente implementar la red sensorial, en la cual se implementó el producto de este trabajo de graduación.

En Guatemala se pueden encontrar ejemplos puntuales de proyectos de monitoreo de signos vitales en los que esta tecnología implementaría una mejora notable, como lo son en los hospitales públicos, donde el tiempo de espera para realizar la medición completa de signos vitales puede ser mayor a una media hora desde el punto en que se es atendido. Este proyecto, en general, busca reducir este tiempo de espera sin afectar la exactitud de las mediciones. También, como se mencionaba anteriormente, puede ser utilizado para un ambiente personalizado, en el cual una persona capacitada puede colocar adecuadamente todos los sensores y en cuestión de minutos obtener los resultados de cada medición. Esto brindaría la facilidad de mantener el control periódico del usuario para posteriores análisis.

V. DELIMITACIÓN E IMPACTO DEL TEMA

La red sensorial donde se implementó la central de información, sistema desarrollado en este trabajo de graduación, se considera el precursor en esta línea de investigación, tanto en Guatemala como en la Universidad del Valle y a pesar de que se han encontrado dispositivos similares en estudios internacionales, ninguno tiene el enfoque presentado por éste ni la cantidad de sensores a implementar. Es decir, se han encontrado proyectos que monitorean uno o dos signos vitales, pero un sistema con monitoreo de más de dos signos vitales y envío de mensajes de texto informativos y de alerta se desconocen.

Por parte de este trabajo de graduación, los alcances consisten en iniciar y mantener una conexión Bluetooth entre el controlador y cada uno de los seis sensores para consultar a cada sensor y así obtener la información importante de cada signo vital. Adicionalmente, el controlador iniciará y mantendrá comunicación Bluetooth con la aplicación en el teléfono celular para obtener los parámetros normales del paciente y enviar los resultados de las mediciones de los sensores. La comunicación Bluetooth está limitada a ser del servicio Bluetooth RFCOMM, simulando un puerto RS-232. El módulo también se encuentra limitado por la cantidad de dispositivos a conectar pues el tipo de conexión Bluetooth sólo permite la conexión entre 7 dispositivos - 1 maestro, en este caso el controlador, y 6 esclavos, en este caso los sensores.

Considerando los alcances mencionados, la central de información es de suma importancia para la red sensorial que se ha mencionado. Esta importancia se debe a dos razones, primero a la necesidad de comunicación entre dispositivos y segundo por el análisis de información requerido. La comunicación entre los sensores y esta central permite unificar la información para su clasificación. Además, constantemente revisa que las conexiones funcionen correctamente para evitar la pérdida de información. En caso se interrumpa la comunicación con alguno de los sensores o con el teléfono celular, se avisará al usuario. La segunda parte tomará la información de los sensores y la colocará en un formato fácil de analizar por el teléfono celular. Si se

excluye la parte de comunicación se tendrían únicamente sensores por separado, cada uno realizando mediciones independientes con lo cual se pierde de vista el objetivo del proyecto. De igual manera, si se elimina el análisis de información, el teléfono celular obtendría datos en todo tipo de formatos, lo cual retrasaría el proceso de análisis de información y podría resultar en datos erróneos con lo cual el médico o cuidador del paciente recibiría valores que tal vez no sean trascendentales o provoquen un diagnóstico equivocado. Así el módulo central de información busca unificar las mediciones de los signos vitales y enviarlas de manera ordenada para facilitar su análisis. Por estas razones se ha tratado tanto la red sensorial, pues aunque no es el propósito de este trabajo de graduación, la implementación final de la central de información necesitó de las demás partes de la red para funcionar completa y correctamente.

VI. METODOLOGÍA

El diseño e implementación de la central de información requirió realizar varios experimentos con el propósito de comprender el funcionamiento de los dispositivos de comunicación Bluetooth para establecer comunicación entre los microcontroladores representantes del control y sensores. La metodología seguida para estos experimentos consistió en plantear el objetivo de la prueba, determinar qué dispositivos se iban a utilizar y diseñar e implementar el programa en el microcontrolador. A continuación se describen tres experimentos cuyos resultados permitieron plantear la topología final de la red. Por último se describe el algoritmo implementado y los resultados obtenidos al armar la red con comunicación hacia los sensores y el teléfono celular. Además se incluye una discusión que analiza la importancia de todos estos resultados y su importancia en el diseño final así como las conclusiones del proyecto.

VII. COMUNICACIÓN ENTRE DOS MÓDULOS BLUETOOTH (RBT-001)

A. DISEÑO

El objetivo de este experimento era comunicar dos módulos Bluetooth Robotech RBT-001. Para alcanzar dicho objetivo se implementaron dos programas en lenguaje ensamblador para comunicar dos microcontroladores. El primer programa actúa como maestro de la comunicación y se utiliza la configuración de Modo Transparente de operación. El segundo programa actúa como esclavo de la comunicación buscando al dispositivo maestro y solicitando la comunicación. Para lograr este tipo de conexión, fue necesario conocer previamente la dirección Bluetooth única del dispositivo maestro. Esto se logró utilizando una computadora para realizar una búsqueda de dispositivos Bluetooth, identificar el módulo RBT-001 y anotar su dirección. Una vez establecida la comunicación, el esclavo le envía información de un contador al maestro para que este lo despliegue en una serie de LED's. Es importante notar que como el maestro funciona en Modo Transparente, éste nunca solicita la información del esclavo, sólo despliega el dato recibido.

1. Diseño del Programa Maestro

a) Inicio
   1) Se configura el reloj interno, el puerto serial con un baud rate de 9600, el módulo Bluetooth, y los puertos de entrada y salida del microcontrolador.

b) Ciclo principal del programa.
   1) Revisa si se recibió un dato por comunicación serial.
   2) Si se recibió un dato:
      i) Despliegue en LEDs del dato.
      ii) Clear del bit de recepción serial.
   3) Si NO se recibió un dato:
      i) Parpadeo de un LED para mostrar cuando no se reciben datos.

2. Diseño del Programa Esclavo

a) Inicio

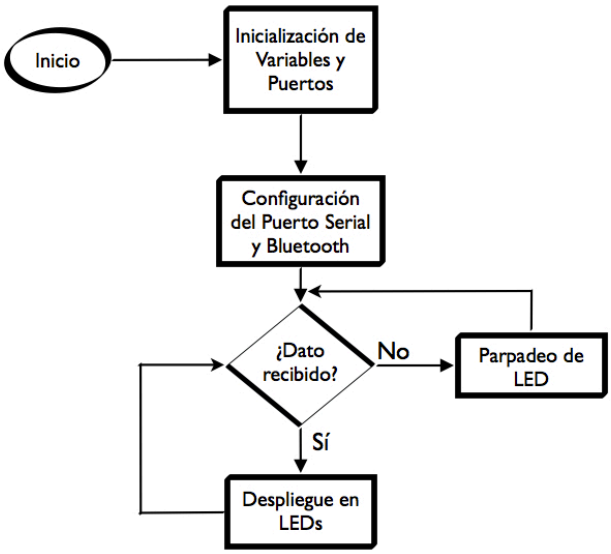
1) Se configura el reloj interno, el puerto serial con un baud rate de 9600 y los puertos de entrada y salida del microcontrolador.
2) Configuración del Módulo Bluetooth
3) Indicar que se desea establecer conexión con el dispositivo maestro.
4) Esperar 5 segundos para que el dispositivo realice la conexión.
5) Establecer el Modo Transparente del RBT-001.
6) Esperar 3 segundos

b) Ciclo principal.
   1) Envío serial del contador.
   2) Incremento del contador.
   3) Retraso de 3ms.

B. RESULTADOS

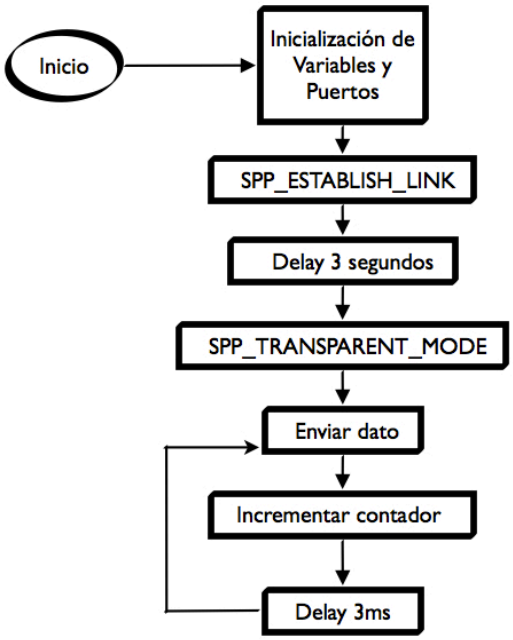
El objetivo del experimento se alcanzó exitosamente pues se lograron comunicar dos microcontroladores utilizando la tecnología Bluetooth por medio de los módulos RBT-001. Se implementaron dos programas en lenguaje ensamblador, uno para el maestro controlador y otro para el sensor esclavo. Para el dispositivo maestro, se configuró el módulo Bluetooth para que funcionara en el modo transparente y actuara como un cable de comunicación RS-232 (full cable replacement). El programa únicamente despliega en LED's el dato recibido del módulo RBT-001. En caso haya instantes donde el microcontrolador no reciba ningún dato, un LED parpadea (Parallax, Inc., 2010). La Figura 4 muestra el diagrama de flujo de este programa.

Figura 4: Diagrama de flujo del programa maestro.



Para realizar exitosamente la conexión se utilizó una computadora para averiguar las direcciones Bluetooth de ambos dispositivos RBT-001 y se encontró que el dispositivo maestro tiene la dirección 00-17-a0-01-63-18 y el dispositivo esclavo tiene la dirección 00-17-a0-01-62-e4. También se implementó un programa en lenguaje ensamblador para el microcontrolador esclavo de la comunicación. En este programa se utilizaron dos configuraciones para el módulo Bluetooth. La primera configuración utiliza la instrucción SPP_ESTABLISH_LINK para establecer el enlace del puerto serial Bluetooth y debe especificarse la dirección del dispositivo con el cual se desea comunicar. Antes de continuar el programa se espera 5 segundos para que el dispositivo busque al maestro y se establezca la conexión. Luego se configuró nuevamente el módulo para establecer el modo transparente utilizando la instrucción SPP_TRANSPARENT_MODE que mantiene la conexión entre los dispositivos pero permite que el flujo de datos sin utilizar el framing para la comunicación entre el RBT-001 y el microcontrolador. La Figura 5 muestra el diagrama de flujo de este programa. Luego realizar la configuración de la comunicación, el programa entra en el ciclo principal donde envía el valor de un contador al microcontrolador maestro y lo incrementa.

Figura 5: Diagrama de flujo del programa esclavo.



VIII. USO DEL MODO COMANDO PARA EL PROGRAMA MAESTRO

A. DISEÑO

El propósito de este experimento fue conectar dos dispositivos Bluetooth, pero cambiando la configuración del maestro para que este pudiera controlar el enlace Bluetooth y tuviera la capacidad de conectarse con hasta 6 dispositivos. Al igual que en el capítulo anterior, se implementaron dos programas en lenguaje ensamblador para comunicar dos microcontroladores. En este caso, se implementó la comunicación full-dúplex y se cambió la configuración del maestro y esclavo. El primer programa actúa como esclavo de la comunicación y se utiliza la misma configuración mencionada en la sección anterior. El segundo programa actúa como maestro de la comunicación buscando al dispositivo esclavo y solicitando la comunicación. Una vez establecida la conexión, ambos microcontroladores envían y reciben información.

1. Diseño del Programa Maestro

a) Inicio
   1) Se configura el reloj interno, el puerto serial con un baud rate de 9600 y los puertos de entrada y salida del microcontrolador.
   2) Configuración del Módulo Bluetooth.
   3) Indicar que se desea establecer conexión con el dispositivo esclavo.
   4) Esperar 5 segundos para que el dispositivo realice la conexión.

b) Ciclo principal.
   1) Envío del contador utilizando el comando SPP_SEND_DATA.
   2) Incremento del contador.
   3) Despliegue de los datos recibidos de forma serial.

2. Diseño del Programa Esclavo

a) Inicio
   1) Se configura el reloj interno, el puerto serial con un baud rate de 9600 y los puertos de entrada y salida del microcontrolador.

2) Configuración del Módulo Bluetooth utilizando el comando SPP_TRANSPARENT_MODE.

b) Ciclo principal.

1) Envío serial del contador.
2) Incremento del contador.
3) Despliegue de los datos recibidos de forma serial.

C. RESULTADOS

Para alcanzar exitosamente los objetivos de este experimento fue necesario estudiar detalladamente los comandos SPP_SEND DATA, SPP_INCOMING_DATA y SET_EVENT_FILTER para crear una Piconet con los distintos módulos del proyecto. La Figura 6 muestra las especificaciones del comando SPP_SEND_DATA. En este comando es muy importante especificar el puerto RFCOMM que se utilizará para que la comunicación sea exitosa.

Figura 6: Detalles del comando SPP_SEND_DATA.

Description	Send data on a SPP link to remote Bluetooth device	
PacketType	REQ	
Opcode	SPP_SEND_DATA	
DataLength	3 + <PayloadSize>	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	PayloadSize 2 bytes	Number of data bytes to send. Valid range is 1 to 330 bytes..
	PayloadData <PayloadSize> bytes	The data to send.

(Parallax, Inc., 2010)

La Figura 7 muestra las especificaciones del comando SPP_INCOMING DATA. Este comando es importante porque no se genera desde el microcontrolador sino que lo envía automáticamente el RBT-001 cada vez que recibe un dato. El microcontrolador recibe todos los 11 bytes y debe determinar cuál byte contiene la información enviada del otro dispositivo y cuáles bytes pertenecen al framing del RBT-001.

Figura 7: Especificaciones del comando SPP_INCOMING_DATA.

Description	Incoming data on a DLC link, from a remote Bluetooth device	
PacketType	IND	
Opcode	SPP_INCOMING_DATA	
DataLength	3 + <PayloadSize>	
Data	LocalPort 1 byte	Local RFCOMM port number. Range 1-30
	PayloadSize 2 bytes	Number of data bytes to send. Valid range is 1 to 330 bytes..
	PayloadData <PayloadSize> bytes	The data to send.

(Parallax, Inc., 2010)

El comando SET_EVENT_FILTER fue utilizado (ver Figura 8) para evitar que el RBT-001 respondiera a cada comando del microcontrolador. Sin embargo, en esta etapa se descubrió que es necesario cambiar el parámetro para que no funcione en el modo "full cable replacement" sino que el RBT-001 envíe indicadores y confirmaciones al microcontrolador.

Figura 8: Especificaciones del comando SET_EVENT_FILTER.

Description	This command is used to set the event filter. The setting is stored in NVS.	
PacketType	REQ	
Opcode	SET_EVENT_FILTER	
DataLength	2	
Data	Filter 1 byte	0x00: All events reported 0x01: No ACL Link Indicators (default) 0x02: No events reported, UART break still generated and detected. 0x03: No events generated, UART break not generated or detected (full cable replacement)

(Parallax, Inc., 2010)

Además de los comandos, fue necesario estudiar las configuraciones del dispositivo para comprender cuándo actúa este como maestro o esclavo de la conexión mostrado en la Tabla I. En ellas se observa que para crear una Piconet, es necesario que al encender el dispositivo, éste sea el que realice las conexiones hacia los demás dispositivos. Además, la tabla muestra que al ser maestro de una Piconet con hasta 7 esclavos, no se puede utilizar el modo transparente de operación, por lo que siempre se usará el framing para enviar y recibir datos con el microcontrolador. La figura también muestra la columna describiendo el funcionamiento de un Esclavo Transparente, el cual será el caso de todos los microcontroladores, a excepción del microcontrolador de control y monitoreo, y del teléfono celular. De esta columna lo

más importante es que sí es posible utilizar el modo transparente para la comunicación entre el microcontrolador y el RBT-001.

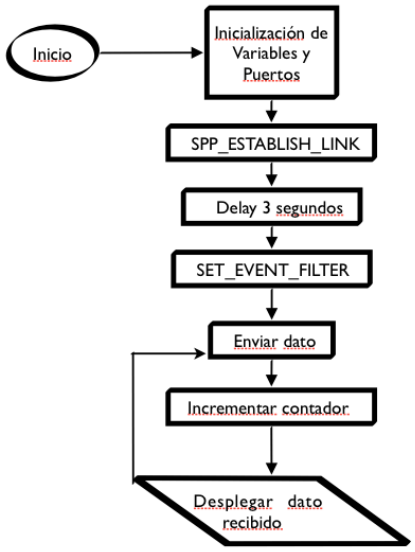
Tabla I: Características del maestro Piconet y esclavo transparente del RBT-001.

Parameter / State	Piconet Master	Transparent Slave
UART Mode	Command	Transparent
Automatic Operation	-	-
Discoverable[1]	yes	no
Connectable[2]	yes	no
Bluetooth Role	Master to x Slaves[3]	Slave to 1 Master
Possible to search for devices (Inquiry)	yes	no
Connect to remote devices (actively)	yes[5]	no
Send Raw Data	no	yes
State after incoming link	Scatternet Master	-
State after outgoing link	Piconet Master	-
State after sending "Transparent Mode" [6]	Transparent Master	-
State after UART BREAK	-	Single Slave

(Parallax, Inc., 2010)

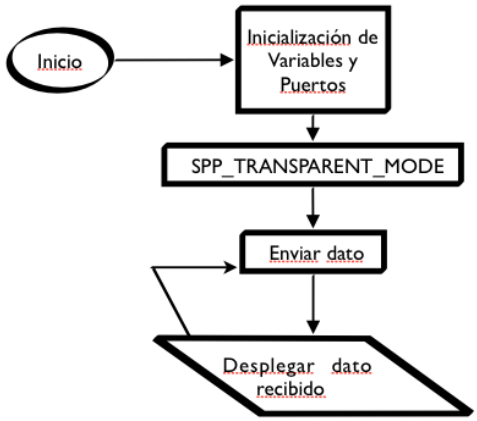
Con todo esto comprendido, se implementó el programa en lenguaje ensamblador para el microcontrolador que actuará como Maestro de la comunicación (ver Figura 9). Utilizando el framing del RBT-001, se utilizó el comando SPP_ESTABLISH_LINK de la misma manera que se utilizó en el experimento anterior para realizar la conexión con el otro módulo Robotech, y además se utilizó el comando SET_EVENT_FILTER para indicarle al módulo Robotech que sí debe comunicarse con el microcontrolador. Luego de realizar la conexión, el programa envía el valor de un contador por medio del Bluetooth y despliega el valor recibido en el puerto B.

Figura 9: Diagrama de flujo del programa maestro.



Por último, también se implementó un programa en lenguaje ensamblador para el microcontrolador Esclavo de la comunicación (ver Figura 10). En este programa se utilizaron dos configuraciones para el módulo Bluetooth. La primera configuración utiliza la instrucción SET_EVENT_FILTER para configurar el módulo a "full cable replacement" utilizado con anterioridad. Luego se envió el comando SPP_TRANSPARENT_MODE para mantener la conexión SPP con el otro dispositivo pero sin tener que utilizar el framing para comunicarse con el RBT-001. Luego de realizar estas configuraciones, se entra en un ciclo que envía el valor de una variable utilizando RS-232 y despliega en el puerto B el valor recibido por la interfaz serial.

Figura 10: Diagrama de flujo del programa esclavo.



IX. COMUNICACIÓN ENTRE EL TELÉFONO CELULAR Y DOS MÓDULOS BLUETOOTH

A. DISEÑO

Este experimento tenía dos objetivos, primero abrir los puertos del RFCOMM del dispositivo maestro para poder establecer más de una conexión y segundo, establecer una conexión exitosa entre el teléfono celular, el microcontrolador maestro y el "sensor" esclavo. Para alcanzar el primer objetivo, se estudiaron detenidamente las características del Serial Port Profile en la hoja de datos de los dispositivos RBT-001 (Parallax, Inc., 2010) para comprender cómo configurar los puertos del RFCOMM. Para comprobar que en efecto ambas conexiones, con el celular y con el sensor, se realizaran exitosamente, se monitoreó el puerto serial de entrada al microcontrolador maestro. Una vez alcanzado este objetivo, se configuró el dispositivo maestro para permitir estas conexiones y se realizaron pruebas para constatar que el segundo objetivo también fuera alcanzado.

B. RESULTADOS

Para alcanzar el primer objetivo, se estudió el manual del dispositivo Robotech RBT-001 (Parallax, Inc., 2010). El dato más importante encontrado fue la instrucción SET_PORTS_TO_OPEN (ver Figura 11) que permite abrir hasta 30 puertos RFCOMM distintos para la comunicación del módulo RBT-001 con otros dispositivos. En el caso del módulo de control, se necesitan tener 6 puertos abiertos, 1 para la comunicación con el teléfono celular y 5 para la comunicación con cada uno de los sensores.

Figura 11: Detalle de la instrucción SET_PORTS_TO_OPEN.

PacketType	REQ	
Opcode	SET_PORTS_TO_OPEN	
DataLength	4	
Data	PORTS 4 Bytes	This field is a 32-bit mask indicating which RFCOMM ports the RBT-001 has to open. Bit 30 and 31 must be set to 0. Bit 0 is RFCOMM port 1 and bit 29 is port 30 e.g. if this field has the value 0x00000007, port 1 to 3 will be opened. All other ports will be closed if open.

(Parallax, Inc., 2010)

El segundo dato importante encontrado en el manual del Robotech RBT-001 fue la existencia de una memoria EEPROM a la cual se refieren como NVS (Non-Volatile Storage por sus siglas en inglés). El manual contiene una tabla donde presenta todas las localidades de la memoria y la información que se guarda en cada localidad. En la Figura 12 se muestra un extracto de esta tabla donde se encontró la localidad de memoria donde se guarda la configuración de los puertos que el dispositivo abre al encenderse. Entonces, para corroborar que la instrucción SET_PORTS_TO_OPEN guardara la configuración de los puertos, se apagó el dispositivo y, al encenderlo nuevamente, utilizando la instrucción READ_NVS se leyeron 4 bytes de la localidad 0x56 para obtener la configuración de los puertos. La respuesta de esta lectura se muestra en la tabla II, indicando que en efecto, se están habilitando los 7 puertos necesarios para la red. Una observación importante es que estos 7 puertos sólo es necesario habilitarlos en el módulo central de monitoreo puesto que al establecer enlaces SPP, los dispositivos RBT-001 permiten que el puerto local sea distinto al puerto remoto.

Figura 12: Extracto de la tabla de localidades NVS.

No.	Address	Parameter	Default Value	Description	SW Reset required
11	0056	SppPorts-ToOpen	0x00000001	Bitmask defining the RFCOMM channels to open. For each channel one RFCOMM instance will be created.	no

(Parallax, Inc., 2010)

Tabla II: Resultados de la lectura de la memoria NVS del módulo maestro.

Start	Tipo de Paquete	Opcode	Data	Length	Checksum	Paquete de datos	End	Descripción
0x02	0x69	0x25	0x05	0x00	0x93	0x04 0x30 0x32 0x31 0x32	0x03	Después del Reset, el RBT-001 está listo para recibir comandos.
0x02	0x43	0x72	0x08	0x00	0xBD	0x00 0x56 0x00 0x04 0x01 0x00 0x000x00	0x03	Lectura sin error de la localidad 0x0056. Leyó 4 bytes 0x00 00 00 01

Una vez se configuraron los puertos en el módulo de monitoreo central, se realizaron pruebas para enlazar al teléfono celular con el módulo central y al módulo central con el otro microcontrolador. La Tabla III muestra una secuencia de bits recibida en la última prueba realizada donde se conectaron los tres dispositivos en una misma red. Se colocó únicamente esta tabla puesto que al conectar los 3 dispositivos se encuentran los mismos datos que al conectar cada uno por separado. La Tabla III muestra, separado por columnas, el significado de cada byte obtenido y una breve descripción de la función del paquete entero siguiendo el framing mencionado anteriormente y mostrado en la Figura 3. De esta manera se explica cómo el módulo central de monitoreo inicia la conexión con el otro módulo RBT-001 utilizando el puerto local 0x02 y el puerto remoto 0x01 y cómo se establece esta conexión exitosamente. También se ve como, antes de que el celular se conecte al módulo, la central intenta enviarle datos pero no hay conexión. Luego, se realiza exitosamente la conexión con el teléfono celular con el puerto local 0x01 y el puerto remoto 0x01. Una vez establecida la conexión con el teléfono, el envío y recepción de datos ocurre sin errores tanto con el teléfono como con el otro módulo RBT-001.

Además de la Tabla III se incluyen abajo las Figuras 13, 14 y 15 que muestran fotografías de la aplicación en el teléfono celular y la recepción de datos del módulo RBT-001 esclavo. La Figura 13 muestra un resultado muy importante. Al momento de tomar la fotografía, se tenían 3 dispositivos Bluetooth, aparte del teléfono celular, encendidos: una computadora personal, el módulo RBT-001 maestro y el módulo RBT-001 esclavo. Sin embargo, cuando el celular hizo la búsqueda de dispositivos únicamente encontró 2: la computadora (Dirección Bluetooth 00-25-00-5E-C5-60) y el módulo RBT-001 maestro (Dirección Bluetooth 00-17-a0-01-63-18). Esto comprueba que la Piconet creada entre el maestro, esclavo y celular, funciona correctamente puesto que el esclavo no debe ser un dispositivo Bluetooth visible y que permita conexiones (ver Tabla I). A pesar de que este resultado se documentó con la aplicación Java del celular, los mismos resultados se obtuvieron al probar con la aplicación Android. La Figura 14 muestra una fotografía de la aplicación cuando recibió el caracter ‘J’ del módulo maestro luego de que se realizara la conexión

Bluetooth. De manera similar la Figura 15 muestra los LED's del puerto B del dispositivo esclavo donde este recibe el byte '01001010', binario del caracter 'J', del dispositivo maestro.

Tabla III: Conexión, envío y recepción de datos entre el módulo maestro, el esclavo y el teléfono celular.

Start	Tipo de paquete	Opcode	Data	Length	Checksum	Paquete	End	Descripción
0x02	0x43	0x0A	0x02	0x00	0x4F	0x00 0x02	0x03	Inicia el intento de conexión SPP en el puerto 0x02.
0x02	0x69	0x3E	0x04	0x00	0xAB	0x02 0x0C 0x00 0x00	0x03	Cambia el estado del puerto SPP 0x02
0x02	0x69	0x0B	0x09	0x00	0x7D	0x00 0xE4 0x62 0x01 0xA0 0x17 0x00 0x02 0x01	0x03	Establece el link SPP exitosamente con el otro RBT-001.
0x02	0x69	0x10	0x04	0x00	0x7D	0x02 0x01 0x00 0xF0	0x03	SPP_INCOMING_DATA: Puerto Local 0x02, Puerto Remoto 0x01
0x02	0x43	0x0F	0x02	0x00	0x54	0x1F 0x01	0x03	SPP_SEND_DATA al puerto 0x01 pero da el error 0x1F que indica que no hay conexión establecida en ese puerto.
0x02	0x43	0x0F	0x02	0x00	0x54	0x00 0x02	0x03	SPP_SEND_DATA por el puerto 0x02 sin error.
0x02	0x69	0x10	0x04	0x00	0x7D	0x02 0x01 0x00 0xF0	0x03	SPP_INCOMING_DATA: Puerto Local 0x02, Puerto Remoto 0x01
Continúa enviando datos primero al puerto 0x01 con error 0x1F y luego al puerto 0x02 sin error. Recibe por el puerto 0x02 el valor 0xF0								
0x02	0x69	0x0C	0x07	0x00	0x7C	0x1E 0xC7 0xF0 0x1A 0x4E 0x44 0x01	0x03	Conexión SPP solicitada remotamente fue establecida correctamente en el puerto 0x01
0x02	0x43	0x0F	0x02	0x00	0x54	0x1E 0x02	0x03	SPP_SEND_DATA por el puerto 0x02 con el error ERROR_CURRENTLY _NO_BUFFER
0x02	0x69	0x3E	0x04	0x00	0xAB	0x01 0x8C 0x00 0x00	0x03	Cambia el estado del puerto SPP 0x01 para comunicación con celular. En Alto: DSR, CTS, DLC.
0x02	0x43	0x0F	0x02	0x00	0x54	0x00 0x01	0x03	SPP_SEND_DATA por el puerto 0x01 sin error.
Recibe datos del celular una sola vez. Envía constantemente datos al celular por el puerto 0x01, y al otro módulo por el puerto 0x02 sin error. Recibe del otro modulo datos sin error.								

Figura 13: Dispositivos encontrados por la aplicación Java del teléfono celular.

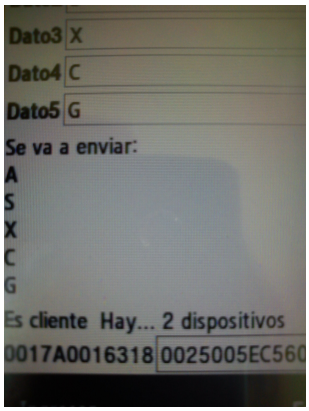


Figura 14: Datos recibidos por la aplicación Java del teléfono celular vía Bluetooth.

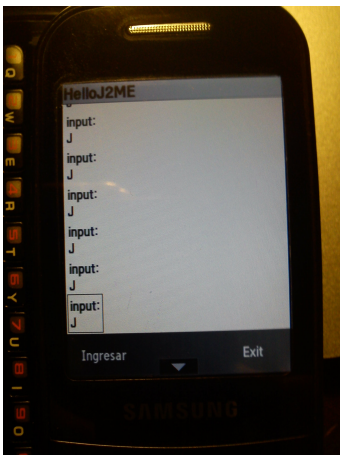
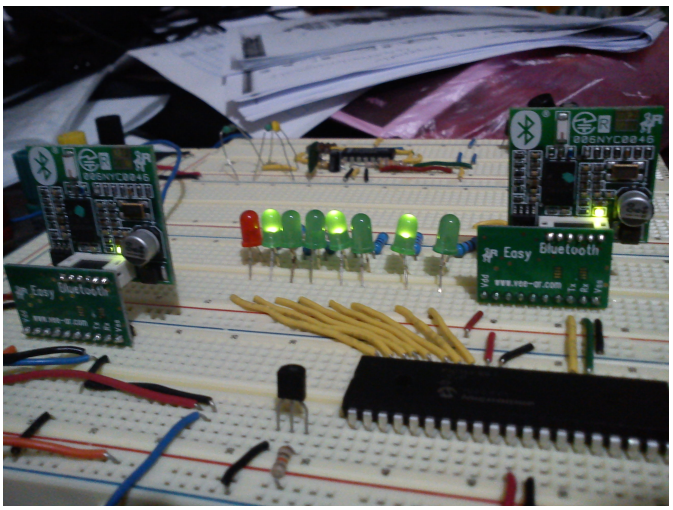


Figura 15: Dato recibido del módulo esclavo vía Bluetooth por el módulo maestro.

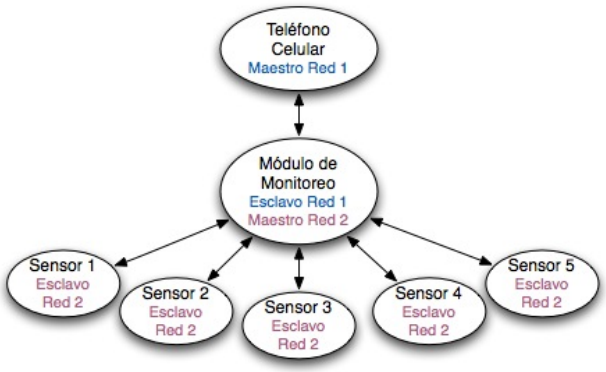


X. IMPLEMENTACIÓN FINAL DE LA RED SENSORIAL

A. DISEÑO

Para lograr la implementación exitosa de la red sensorial fue necesario definir dos aspectos primordiales: la topología de la red y el algoritmo para el programa del microcontrolador maestro. La topología de la red se definió con base en el funcionamiento de los módulos RBT-001 en modo comando utilizando como base los resultados descritos en el capítulo "Comunicación entre el Teléfono Celular y dos Módulos Bluetooth". De acuerdo con estos criterios, se necesitaba una topología que permitiera al maestro de la red obtener los datos de cada uno de los sensores y enviarlos al teléfono celular. La topología escogida fue la mostrada en la Figura 16, la cual muestra una topología de árbol que permite realizar un barrido de revisión de los sensores, para luego empaquetar la información y enviarla al teléfono.

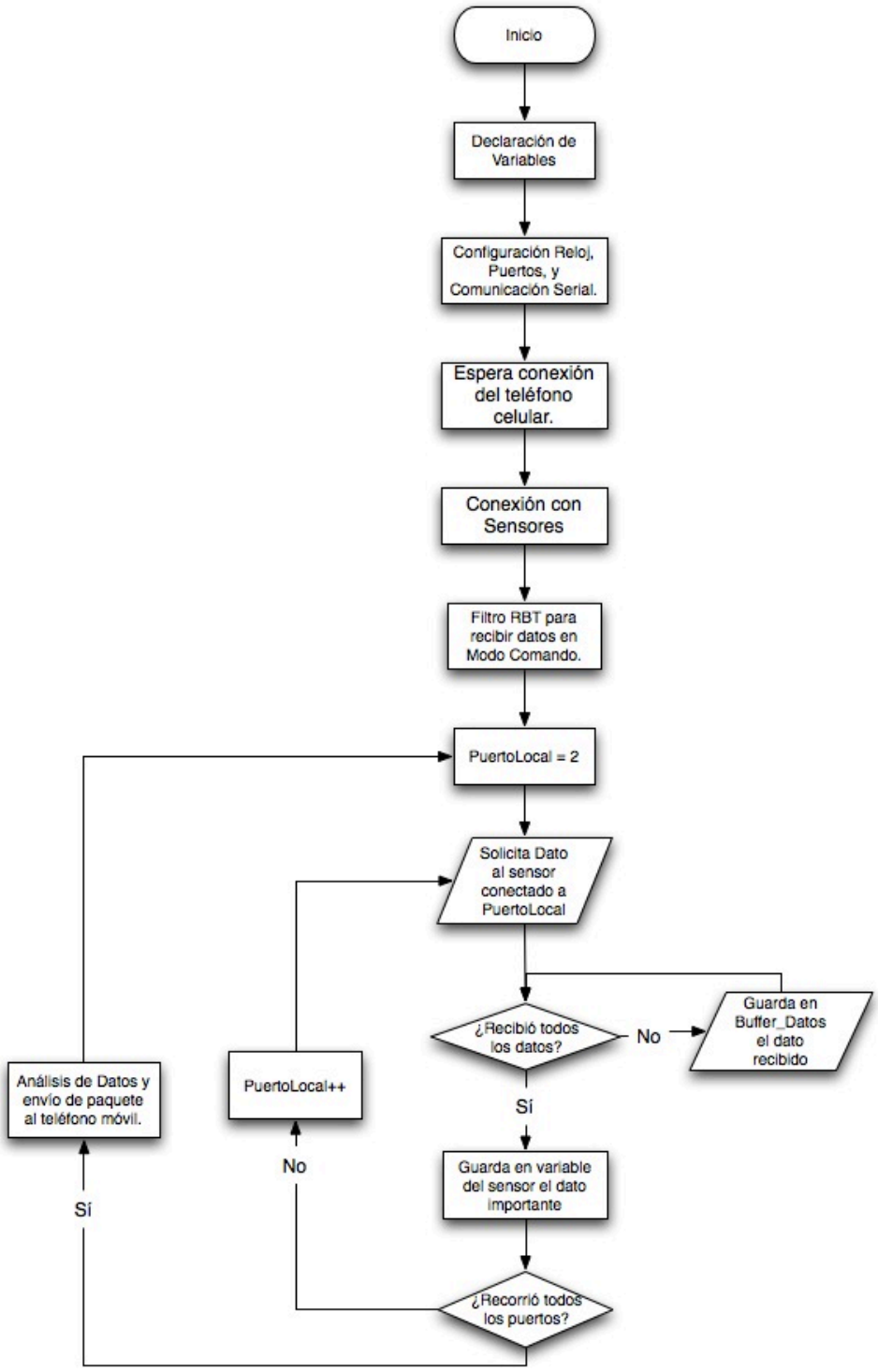
Figura 16: Topología de la red de comunicación.



De acuerdo con la topología, también se diseñó el algoritmo del microcontrolador de monitoreo que revisará cada sensor para obtener la información. El diagrama del algoritmo a implementar se muestra en la Figura 17. Una de las partes cruciales del algoritmo se encuentra en la conexión con el teléfono celular. Debido al tipo de implementación que será la red sensorial, se decidió que la mejor opción era esperar una conexión entrante del teléfono antes de iniciar la conexión con los sensores y la obtención de datos. Adicionalmente, es importante mencionar que después de cada conexión el programa revisa que la conexión sea exitosa, y de no

serlo detiene el flujo del programa. Este programa fue escrito en lenguaje C de alto nivel utilizando la versión Lite del compilador High-Tech C de Microchip [4].

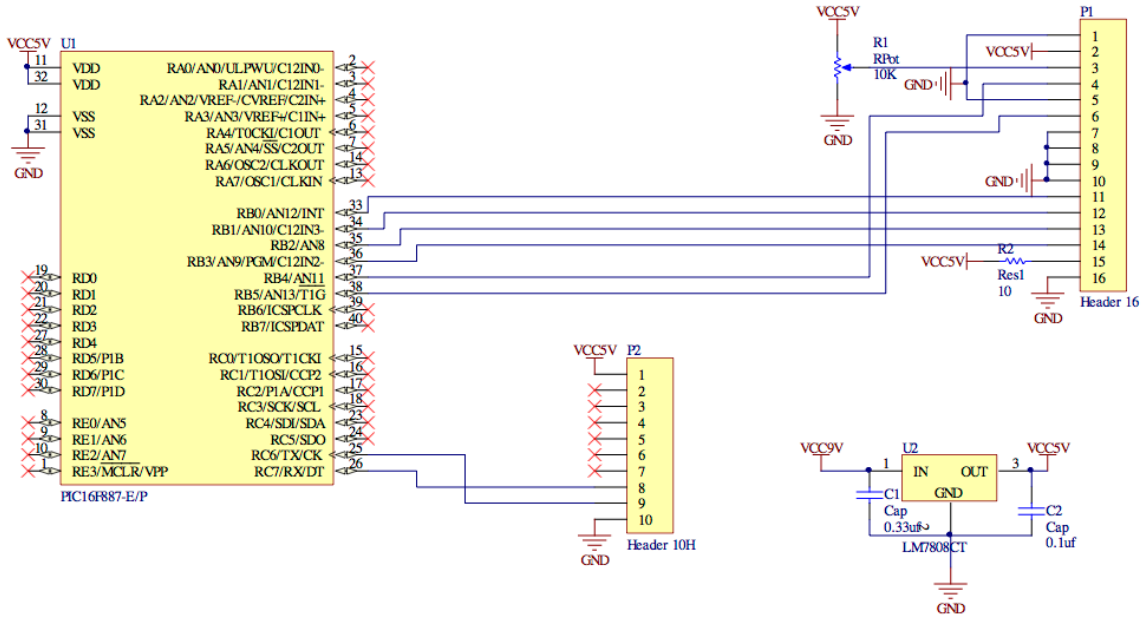
Figura 17: Algoritmo del programa del microcontrolador maestro.



B. RESULTADOS

Siguiendo el diagrama de flujo mostrado en la Figura 16 se programó el microcontrolador 16F887, mismo que se ha utilizado en las secciones anteriores. Adicionalmente, se agregó un regulador de voltaje LM7805 para regular una batería de 9V a los 5V con los que funciona el microcontrolador y también se agregó una pantalla LCD de 2x16 caracteres donde se despliegan mensajes para informar al usuario de las acciones que está ejecutando el programa. El circuito implementado se muestra en la Figura 18.

Figura 18: Circuito del módulo de monitoreo.



Una vez armado este circuito, se inició con la etapa de pruebas de conexión con los otros módulos Bluetooth y con el teléfono celular. En las Figuras 19, 20 y 21 se muestran los mensajes desplegados en la pantalla cuando se configura el módulo RBT-001, espera la conexión del teléfono y se conecta el módulo central al primer sensor.

Figura 19: Mensaje desplegado al ejecutar la instrucción SET_EVENT_FILTER.

Figura 20: Mensaje mostrado cuando el maestro espera la conexión con el teléfono celular.

Figura 21: Mensaje mostrado al iniciar la conexión utilizando la instrucción SPP_ESTABLISH_LINK.

La primera etapa de pruebas incluyó únicamente pruebas de conexión, envío y transmisión de datos de dos sensores de la red. Durante estas pruebas fue necesario realizar una tabla que mostrara todos los posibles errores que pueden ocurrir durante

la conexión para facilitarle al usuario la comprensión cuando algo no funciona correctamente en la comunicación. La Tabla IV se muestra el mensaje que será desplegado en la pantalla y en caso de error, el tipo de error que representa. Cabe destacar que durante todas las pruebas realizadas nunca ocurrió ninguno de estos errores, pues la mayoría ocurren cuando un módulo RBT-001 no funciona correctamente. Sin embargo, para futuros usos es importante conocer cuáles son y cuándo pueden ocurrir.

Tabla IV: Mensajes y códigos de error que pueden ocurrir.

Mensaje	Condición	
	No. Error	**Significado**
Configurando Bluetooth…	--	Inicia la configuración del módulo RBT-001.
Configuración Correcta	0x00	No ocurrió ningún error.
Error No. __	0x01	Error en los parámetros de configuración
	0x1B	Se excedieron los límites en los parámetros de enviados.
Conectando Sensor	--	Inicia la conexión con el sensor indicado.
Conexión Correcta	0x00 y 0x00	No ocurrió ningún error.
Error No. __	0x22	Puerto SPP está ocupado.
	0x21	Puerto SPP no está habilitado.
	0x20	Puerto SPP inválido.
	0x26	Se solicitó iniciar una conexión que ya se estaba realizando.
y __	0x01	No existe conexión DLC.
	0x02	No existe el puerto RF.
	0x03	No se pudo establecer el DLC.
	0x04	No se autorizó acceso al puerto.
	0x05	No existe una comunicación DLC/L2CAP.

Otra observación realizada durante las pruebas se dio cuando se incorporó el teléfono celular a la red. En las primeras pruebas realizadas, se lograba establecer

comunicación con el teléfono, pero luego era imposible establecer comunicación con los sensores. Para solucionar este inconveniente se monitoreó el puerto de recepción serial del microcontrolador y se encontró que cuando el celular se conectaba con el módulo RBT-001, este entraba en Modo Transparente, lo cual causaba que la única comunicación activa se diera entre el teléfono y el módulo, pero este no podía realizar conexiones con los demás sensores. Para eliminar este error fue necesario consultar nuevamente la hoja de datos del funcionamiento del RBT-001. En esta hoja se encontró la Tabla V y VI, las cuales explicaron lo que sucedía al momento de realizar el teléfono la conexión.

Tabla V: Funcionamiento del RBT-001 en modo automático.

Parameter / State	Idle Automatic	Transparent Slave
UART Mode	Command	Transparent
Automatic Operation	0x01 (On)	-
Discoverable[1]	yes	no
Connectable[2]	yes	no
Bluetooth Role	-	Slave to 1 Master
Possible to search for devices (Inquiry)	yes	no
Connect to remote devices (actively)	yes	no
Send Raw Data	-	yes
State after incoming link	Transparent Slave	-
State after outgoing link	Piconet Master	-
State after sending "Transparent Mode" [6]	-	-
State after UART BREAK	-	Single Slave

En la Tabla V se encontró que el módulo RBT-001 está configurado por defecto para funcionar en modo automático, lo cual causa que al conectarse otro dispositivo a él, por ejemplo el celular, éste cambia automáticamente a funcionar en Modo Transparente y le es imposible realizar conexiones a otros dispositivos, según se

puede ver circulado en verde en la Tabla V. Por ende, se investigó más a fondo la hoja de datos para cambiar esta configuración y permitir que el RBT-001 maestro pudiera recibir la conexión del teléfono y conectarse a los sensores. Se encontró que para lograr el funcionamiento deseado era necesario cambiar el RBT-001 maestro del funcionamiento Automático al No-Automático utilizando la función WRITE_OPERATION_MODE (Figura 22) del dispositivo Bluetooth. Una vez realizado este cambio el módulo de monitoreo fue capaz de esperar la conexión del teléfono y, una vez conectados, realizar las conexiones hacia los demás sensores de la red tal como lo muestra la Tabla VI.

Tabla VI: Funcionamiento del RBT-001 en modo no-automático.

Parameter / State	Idle Non-Automatic	Single Slave
UART Mode	Command	Command
Automatic Operation	0x00 (Off)	-
Discoverable[1]	yes	yes
Connectable[2]	yes	yes
Bluetooth Role	-	Slave to 1 Master
Possible to search for devices (Inquiry)	yes	yes
Connect to remote devices (actively)	yes	yes[4]
Send Raw Data	-	no
State after incoming link	Single Slave	Scatternet Slave
State after outgoing link	Piconet Master	Scatternet Master
State after sending "Transparent Mode" [6]	-	Transparent Slave
State after UART BREAK	-	-

Figura 22: Especificaciones del comando WRITE_OPERATION_MODE.

Description	This command will change the operation mode stored in NVS. The new setting will take effect after a reset.	
PacketType	REQ	
Opcode	WRITE_OPERATION_MODE	
DataLength	1	
Data	Mode 1 Byte	0x00 Automatic Operation OFF 0x01 Automatic Operation ON

(Parallax, Inc., 2010)

XI. DISCUSIÓN

Este trabajo de graduación se enfocó en la implementación de una red de comunicación inalámbrica que comunica 6 sensores de signos vitales con la central de información y ésta a su vez se conecta con el teléfono celular. Para lograr una implementación adecuada de dicha red se escogió la tecnología Bluetooth como interfaz de comunicación, se realizaron varios experimentos para determinar la mejor implementación de la red y por último se incluyó una pantalla LCD para mostrar al usuario mensajes acerca del funcionamiento de la red y los datos obtenidos de los sensores. De esta manera se logró implementar la red propuesta y alcanzar los objetivos individuales establecidos.

Previo al diseño de la red fue necesario estudiar y elegir el medio por el cual se comunicarían los dispositivos. El medio por preferencia y simplicidad física para la conexión de dispositivos fue el medio inalámbrico. Una vez elegida esta opción, se estudiaron distintas tecnologías de comunicación inalámbrica superficialmente, y se escogió la tecnología Bluetooth desde el inicio por haber sido diseñada para la comunicación en redes con pocos dispositivos. Seguidamente se buscaron dispositivos que permitieran la comunicación con esta tecnología y los cuales tuvieran una interfaz sencilla. Esta búsqueda resultó en el dispositivo RBT-001 ampliamente descrito a lo largo de este trabajo. Una vez encontrado éste no se consideró necesario realizar alguna comparación con otros dispositivos o continuar la búsqueda pues este cumplía los requerimientos. Incluso, a pesar de que en efecto se encontraron otros dispositivos no se incluyeron sus características o comparaciones pues ningún otro presentó la facilidad de interfaz con microcontroladores como lo hizo el RBT-001. Esto llevó a la conclusión de que las comparaciones realmente no tenían sentido alguno pues al ser Bluetooth un estándar, todos los dispositivos tenían similares características a excepción de la interfaz de comunicación RS-232.

Una vez establecido el medio se realizó la compra de los dispositivos RBT-001 y se inició con la etapa de pruebas. De todas las pruebas y experimentos realizados, prevalecen los resultados de los experimentos mencionados en este trabajo. El primer

experimento fue una exploración del funcionamiento de los módulos y sus resultados justifican la facilidad de comunicación entre el microcontrolador y el RBT-001 mencionada con anterioridad. Los siguientes experimentos documentan el proceso de diseño de la red implementada y demuestran cómo se llegó al resultado final. Una parte primordial en el diseño de la red fue la parte del monitoreo del puerto serial de recepción del microcontrolador. Esta etapa se considera de suma importante pues permitió generar las Tablas II, III, IV que presentan las respuestas del RBT-001 a los comandos de conexión descritos en los resultados. De estas pruebas se desarrolló el algoritmo de implementación final pues se conoce cómo va a responder el RBT-001 a cada configuración solicitada. Las últimas pruebas y experimentos fueron dedicados a unificar los sensores a la red y comprobar que la central de información funcionara correctamente utilizando la interfaz de la LCD. También en esta última etapa se realizaron pruebas con una aplicación Android del teléfono celular. De todas las 20 pruebas realizadas, no se encontró pérdida de datos, sólo se observó un retardo en la conexión cuando habían demasiados dispositivos Bluetooth en el salón de pruebas y se concluye que este retardo en la conexión se debe a que para encontrar el dispositivo solicitado debe revisar cada dirección Bluetooth encontrada.

Además, las pruebas realizadas corroboraron que la topología propuesta fue adecuada para la aplicación propuesta. La primera razón por la cual fue un éxito la topología propuesta fue porque simplificó el proceso de unificación de la red. En la unificación, los sensores tuvieron que incluir dos instrucciones para configurar su módulo RBT-001 en modo transparente y con reemplazo completo de cable para recibir toda la información directamente en RS-232. Esto facilitó inmensamente el trabajo debido a que cada sensor utiliza un lenguaje y flujo de programa diferente, por lo cual utilizar el modo comando hubiera complicado la integración e incluso podría afectar el algoritmo de obtención de la medición. En cambio, utilizando la comunicación RS-232, los microcontroladores no pierden mucho tiempo enviando el dato y pueden dedicarse a realizar las mediciones. Además, para evitar que los sensores tuvieran complicaciones para saber quién se comunica con ellos, cada sensor tiene una única conexión con la central de información y la central es la que controla con quiénes está conectado y quién debe enviarle información. Fue así como la

integración se simplificó y según las pruebas los datos se recibieron dentro del período de tiempo propuesto y sin errores.

Además de la integración de los sensores, fue muy importante corroborar el funcionamiento del algoritmo de la central de información. Esto se logró por medio de las pruebas y utilizando los mensajes desplegados en la pantalla LCD. También se compararon los resultados de la Tabla III con los mensajes desplegados para corroborar que verdaderamente no hubieran errores en la comunicación. Después de las pruebas y de corroborar que los mensajes mostraran los sucesos reales pasando en la comunicación, se concluyó que la implementación del algoritmo fue el adecuado y cumple con los objetivos propuestos. Esta es tal vez la parte que más se podría optimizar pues a pesar de que la implementación actual funciona adecuadamente para la aplicación propuesta, si en algún punto se desea expandir el proyecto, es importante mejorar ciertos aspectos. Uno de los aspectos más importantes a considerar en una futura expansión es el dispositivo Bluetooth RBT-001 debido a la cantidad máxima de dispositivos que se pueden conectar en la red. En la configuración de red actual el máximo número de sensores que se pueden conectar es 7 lo cual deja únicamente 2 sensores que se pudieran agregar en una expansión. Si fuera necesario agregar más, es imperativo cambiar el dispositivo de comunicación inalámbrica, por uno con funcionamiento similar para poder adaptar la topología de la red al nuevo dispositivo y así lograr la expansión. Adicionalmente, debe revisarse las capacidades del microcontrolador implementado para determinar si su capacidad de procesamiento es suficiente para la teórica expansión.

Fue a través de estos experimentos y pruebas que se logró implementar exitosamente la Central de Información. Los resultados de los experimentos muestran el avance progresivo del módulo hasta su implementación final. La pantalla LCD proporciona una interfaz gráfica apropiada para mostrar los mensajes en el avance del programa. Unificando todos estos detalles se entregó la central de información funcional, con comunicación cumpliendo los objetivos y con un éxito excepcional en la transmisión y recepción de datos.

XII. CONCLUSIONES Y RECOMENDACIONES

1. Se implementaron exitosamente dos programas que conectan vía Bluetooth a dos microcontroladores donde el maestro funcionaba en modo comando y el esclavo en modo transparente para simular el funcionamiento de los sensores y el control de la red.
2. Se configuró el módulo maestro para abrir 7 puertos RFCOMM para la red sensorial y comunicación con el teléfono celular.
3. Se diseñó una red con topología de árbol que permite recolectar los datos de los sensores y enviar la información hacia el teléfono celular.
4. Se implementó una pantalla LCD de 2x16 caracteres para desplegar mensajes al usuario acerca del flujo del programa, errores en la comunicación y los datos obtenidos de los sensores debidamente identificados según las mediciones de signos vitales.
5. Se estableció exitosamente la red Scatternet entre el módulo maestro, los 5 sensores de signos vitales y el teléfono celular.
6. Para futuras expansiones se recomienda modificar la topología de la red agregando otro módulo Bluetooth para aumentar la cantidad de sensores que se pueden conectar al sistema.

XIII. BIBLIOGRAFÍA

[1] Bluetooth SIG. (2010). Bluetooth Wireless Technology Profiles. Retrieved 11 20, 2011, from http://www.bluetooth.com/English/Technology/Works/Pages/Profiles_Overview.aspx

[2] Microchip Technology, Inc. (2010). *HI-TECH C for PIC10/12/16.* EEUU.

[3] Parallax, Inc. (2010). *AppNote: Easy Bluetooth to Easy Bluetooth Communication.* EEUU.

[4] Parallax, Inc. (2010). *Bluetooth Serial Module User Manual.* Sarzana, Italia: Robotech srl.

XIV. APÉNDICE

A. PROGRAMA MAESTRO EN PIC C IMPLEMENTADO EN LA CENTRAL DE INFORMACIÓN

```
/*
* Central de Información
* Controla la comunicación entre los 6 sensores y el teléfono
celular
*
* IMPORTANTE: El EasyBT NO debe estar en modo automático
*               Utilizar ReadSetOpMode para quitar modo automático
*
* Juana Rivera 07856
* 25/03/2012
*/

#include <htc.h>
#include "usartMine.c"
#include "lcd.c"
#include <stdlib.h>

__CONFIG (LVP_OFF & FCMEN_ON & IESO_OFF & BOREN_OFF & CPD_OFF &
CP_OFF & MCLRE_ON & PWRTE_ON & WDTE_OFF & FOSC_EXTRC_NOCLKOUT);
__CONFIG (WRT_OFF & BOR4V_BOR21V);

char puertoLocal; //Puerto por el cual se enviaran los datos
char recibido; //dato recibido util
char dato; //Dato a enviar
char longitud; //Longitud de la cadena recibida
int i;//Para el ciclo for
unsigned short puntero=0; //Puntero del buffer de datos recibidos
char contador=0;

char buffer_datos[74]; //Buffer de datos recibidos
char texto[4];

//Direcciones de los modulos bluetooth de cada sensor.
char direcciones[4][6] = { {0x00, 0x17, 0xA0, 0x01, 0x62, 0xE4},
                           {0x00, 0x17, 0xA0, 0x01, 0x6C, 0xD7},
                           {0x00, 0x17, 0xA0, 0x01, 0x6D, 0xA8},
                           {0x00, 0x17, 0xA0, 0x01, 0x6B, 0xD3},
                                   {0,0,0,0,0,0},
                                   {0,0,0,0,0,0}};

char presion[2]; //Presion alta y baja
char pulso; //Pulso cardiaco
char glucosa[3]; //Dato del glucometro
char termometro[4]; //Dato del termometro
char peso[6];
```

```
char respiracion;
char celular[4]; //Eco del celular

//Lista de metodos en el programa
void Setup_BT_SPP_RBT(char address[], char puerto);
void Setup_BT_Filter();
void SPP_Write(char envia, char puerto);

//Contadores para ciclos
char counter =0;
int cont2 = 0;

void main(){

    //Frecuencia Oscilador
    OSCCON = 0b01110101;

    ANSEL  = 0;
    ANSELH = 0;

    init_usart(); //Inicializa el puerto serial con baud rate de
9600
    __delay_ms(10);

    //Configuracion de puertos
    TRISA = 0;
    TRISB = 0;
    TRISD = 0;
    PORTB = 0;
    PORTD = 0;
    PORTA = 0;
    TRISAbits.TRISA2 = 1;

    lcd_init(); //Inicializa pantalla LCD
    lcd_clear();

    //Configura el RBT-001 para recibir datos
    lcd_puts("Configurando");
    lcd_goto(40);
    lcd_puts("Bluetooth...");

    puntero = 0;
    Setup_BT_Filter();

    //Espera y Revisa que se configuro bien el RBT-001
    while(puntero<7){}
    if(buffer_datos[6] != 0 && buffer_datos[14] != 0){
     itoa(texto, buffer_datos[6], 16);
     lcd_clear();
     lcd_puts("Error No. ");
```

```
 lcd_puts(texto);
 while(1){}
}
else{
 lcd_clear();
 lcd_puts("Configuracion");
 lcd_goto(40);
 lcd_puts("Correcta");
}

//*************************************************************
//Espera la conexion con el celular

lcd_clear();
lcd_puts("Esperando");
lcd_goto(40);
lcd_puts("Conexion");

puntero = 0;
while(puntero < 40){}

//Revisa que se conecto correctamente y
//por el puerto correcto (puerto local 1)
if(buffer_datos[31] != 0 && buffer_datos[20] != 1){
 lcd_clear();
 lcd_puts("Error No. ");
 itoa(texto, buffer_datos[31], 16);
 lcd_puts(texto);
 lcd_goto(40);
 lcd_puts("Puerto: ");
 itoa(texto, buffer_datos[20], 16);
 lcd_puts(texto);
      while(1){}
}

else{
 lcd_clear();
 lcd_puts("Conexion exitosa");
 lcd_goto(40);
 lcd_puts("con telefono");
}

//*************************************************************

//*************************************************************
//Conexion con los sensores
for(puertoLocal = 2; puertoLocal<=5; puertoLocal++){

 lcd_clear();
 lcd_puts("Conectando");
```

```
      lcd_goto(40);
      lcd_puts("Sensor ");
      itoa(texto, (puertoLocal-1), 10);
      lcd_puts(texto);

      puntero = 0;
      Setup_BT_SPP_RBT(direcciones[(puertoLocal-2)], puertoLocal);

      //Espera y Revisa que la conexion sea exitosa
      while(puntero<36){}

      if(buffer_datos[6] != 0 || buffer_datos[26] != 0){
            lcd_clear();
            lcd_puts("Errores No.");
            lcd_goto(40);
            itoa(texto, buffer_datos[6], 16);
            lcd_puts(texto);
            lcd_puts(" y ");
            itoa(texto, buffer_datos[26], 16);
            lcd_puts(texto);
           while(1){}
      }
      else{
            lcd_clear();
            lcd_puts("Conexion");
            lcd_goto(40);
            lcd_puts("Correcta");
      }
     }
     //************************************************************

     while(1){

      PORTDbits.RD0 = 0;//LED que parpadea para controlar si se
queda trabado el ciclo

      lcd_clear();
      lcd_puts("Obteniendo");
      lcd_goto(40);
      lcd_puts("Datos...");

      //Manda un numero solicitando datos al sensor 1
      puntero = 0;//Inicio del buffer de datos
      puertoLocal = 2; //Primer sensor
      SPP_Write(0xFF,puertoLocal);

      //Espera a recibir todos los datos por el puerto serial
      while(puntero <= 41){}

      //Guarda los datos
```

```
      presion[0] = buffer_datos[18];
      presion[1] = buffer_datos[29];
      pulso =  buffer_datos[40];

      //Manda un numero solicitando datos al sensor 2
      puntero = 0;//Inicio del buffer de datos
      puertoLocal = 2; //Segundo sensor
      SPP_Write(0xFA,puertoLocal);

      //Espera a recibir todos los datos por el puerto serial
      while(puntero <= 52){}

      termometro[0] = buffer_datos[18];
      termometro[1] = buffer_datos[29];
      termometro[2] = buffer_datos[40];
      termometro[3] = buffer_datos[51];

      //Manda un numero solicitando datos al glucometro
      puntero = 0;
      puertoLocal = 3;
      SPP_Write(0xFF,puertoLocal);

      //Espera a recibir todos los datos por el puerto serial
      while(puntero <= 41){}

      //Guarda el dato del glucometro
      glucosa[0] = buffer_datos[18];
      glucosa[1] = buffer_datos[29];
      glucosa[2] = buffer_datos[40];


      //Manda un numero solicitando datos al sensor de frecuencia
respiratoria
      puntero = 0;//Inicio del buffer de datos
      puertoLocal = 4; //Cuarto sensor
      SPP_Write(0xFF,puertoLocal);

      //Espera a recibir todos los datos por el puerto serial
      while(puntero <= 19){}

      respiracion = buffer_datos[18];

      //Manda un numero solicitando datos al sensor de peso
      puntero = 0;//Inicio del buffer de datos
      puertoLocal = 3;//5; //Segundo sensor
      SPP_Write(0xFF,puertoLocal);

      //Espera a recibir todos los datos por el puerto serial
      while(puntero <= 74){}
```

```
peso[0] = buffer_datos[18];
peso[1] = buffer_datos[29];
peso[2] = buffer_datos[40];
peso[3] = buffer_datos[51];
peso[4] = buffer_datos[62];
peso[5] = buffer_datos[73];

//Despliega lo que recibio
lcd_clear();
lcd_puts("Presion ");
itoa(texto, presion[0], 10);
lcd_puts(texto);
lcd_puts("/");
itoa(texto, presion[1], 10);
lcd_puts(texto);
lcd_goto(40);
lcd_puts("Pulso ");
itoa(texto, pulso, 10);
lcd_puts(texto);

__delay_ms(500);

lcd_clear();
lcd_puts("Temperatura");
lcd_goto(40);
if(termometro[0]==0xFF){
     for(i=0;i<4;i++){
          itoa(texto, termometro[i], 10);
          lcd_puts(texto);
          termometro[i] = 0;
     }
}
else{
     lcd_puts(termometro);
     termometro[0] = 0;
     termometro[1] = 0;
     termometro[2] = 0;
     termometro[3] = 0;
}

/__delay_ms(500);

lcd_clear();
lcd_puts("Glucosa ");

lcd_putch(glucosa[0]);
lcd_putch(glucosa[1]);
lcd_putch(glucosa[2]);

__delay_ms(500);
```

```
lcd_clear();
lcd_puts("Frecuencia R ");
itoa(texto, respiracion, 10);
lcd_puts(texto);


__delay_ms(500);

lcd_clear();
lcd_puts("Peso");
lcd_goto(40);
if(peso[0]==0xFF){
     for(i=0;i<6;i++){
          itoa(texto, peso[i], 10);
          lcd_puts(texto);
          peso[i] = 0;
     }
}
else{
     lcd_puts(peso);
     peso[0] = 0;
     peso[1] = 0;
     peso[2] = 0;
     peso[3] = 0;
     peso[4] = 0;
     peso[5] = 0;
}

__delay_ms(500);

//Manda al celular
puntero = 0;
SPP_Write(presion[0],1);

while(puntero<=8){}//34){}

//celular[0] = buffer_datos[18];

puntero = 0;
SPP_Write(presion[1],1);

while(puntero<=8){}//19){}

//celular[1] = buffer_datos[18];

puntero = 0;
SPP_Write(pulso,1);

while(puntero<=8){}//19){}
```

```
      //celular[2] = buffer_datos[18];

      puntero = 0;
      for(i=0;i<4;i++){
            SPP_Write(termometro[i],1);
            while(puntero<=8){}
            puntero = 0;
      }

      puntero = 0;
      for(i=0;i<3;i++){
            SPP_Write(glucosa[i],1);
            while(puntero<=8){}
            puntero = 0;
      }

      while(puntero<=34){}

      celular[2] = buffer_datos[18];
      celular[3] = buffer_datos[29];

      lcd_clear();
      lcd_puts("Datos Celular");
      lcd_goto(40);

      for(contador=0;contador<4;contador++){

            itoa(texto, celular[contador], 10);
            lcd_puts(texto);
            lcd_puts(" ");
            //__delay_ms(100);

      }

      PORTDbits.RD0 = 1;
      __delay_ms(5000);
     }

}

//Conecta el master con el sensor establecido
//Recibe como parametros la direccion del dispositivo con el que
se quiere
//conectar, y el puerto local donde se realizara la conexion
void Setup_BT_SPP_RBT(char address[], char puerto){
    sendch(0x02); // Start of Transmission
    sendch(0x52); // Type = REQ
    sendch(0x0A); // Opcode = SPP_ESTABLISH_LINK
    sendch(0x08); // Length = 8 (16-bit)
```

```
    sendch(0x00);
    sendch(0x64); // Checksum
    sendch(puerto); //Puerto Local

    //Recorre el arreglo con la direccion y la envia
    //Lo recorre al reves porque asi lo requiere el RBT-001
    for(i=5; i>=0; i--) sendch(address[i]);

    sendch(0x01); // Puerto en dispositivo remoto
    sendch(0x03); // End of Transmission
}

//Define el filtro para el reporte de eventos del RBT
void Setup_BT_Filter(){
    sendch(0x02); // Start of Transmission
    sendch(0x52); // Type = REQ
    sendch(0x4E); // Opcode = SET_EVENT_FILTER
    sendch(0x01); // Length = 1 (16-bit)
    sendch(0x00);
    sendch(0xA1); // Checksum
    sendch(0x01); // Default: No ACL Events Reported
    sendch(0x03); // End of Transmission
}

//Define el filtro para el reporte de eventos del RBT
void SPP_Write(char envia, char puerto){
    sendch(0x02); // Start of Transmission
    sendch(0x52); // Type = REQ
    sendch(0x0F); // Opcode = SPP_SEND_DATA
    sendch(0x04); // Length = 4 (16-bit)
    sendch(0x00);
    sendch(0x65); // Checksum
    sendch(puerto); // Puerto Local
    sendch(0x01); // Cantidad de datos
    sendch(0x00); // Cantidad de datos
    sendch(envia); // Dato
    sendch(0x03); // End of Transmission
}

//Interrupciones
void interrupt interrupciones(void){

    if(RCIF){
     buffer_datos[puntero] = RCREG;
     puntero++;
    }
}
```

XV. GLOSARIO

Baud Rate: cantidad de paquetes enviados por segundo.

Esclavo: en comunicaciones, dispositivo que únicamente responde a las peticiones del maestro, no contribuye información sin que se la soliciten.

Framing: paquete compuesto por varias partes, cada una transmitida individualmente.

Full Cable Replacement: término que describe el estado cuando el RBT-001 no le envía ninguna notificación al microcontrolador, sólo datos.

LED: Light Emitting Diode, diodo emisor de luz.

Lenguaje Ensamblador: lenguage de programación de bajo nivel.

Maestro: en comunicaciones, dispositivo que controla quién envía y recibe información en la red.

Microcontrolador: circuito integrado con las 3 partes básicas de una computadora: memoria, dispositivos de entrada y salida y unidad central de procesamiento.

Modo Comando: modo de funcionamiento del RBT-001 donde la comunicación con éste funciona únicamente por medio de paquetes (framing).

Modo Transparente: en el RBT-001, modo de operación donde el microcontrolador no está conciente de la existencia del mismo.

Piconet: tipo de red Bluetooth donde existe un maestro y hasta 7 esclavos.

READ_NVS: Instrucción del RBT-001 para leer localidades de la memoria EEPROM del dispositivo.

Red Ad-hoc: red inalámbrica descentralizada, es decir todos los dispositivos tienen la capacidad de reenviar la información.

RFCOMM: Comunicación de radio frecuencia que emula el puerto RS-232 con todas las señales de control.

RS-232: Recommended Standard 232, estándar para la transmisión de datos por puerto serial utilizando 15V a 3V para un cero lógico y -15V a -3V para un uno lógico.

SET_EVENT_FILTER: Instrucción RBT-001 para configurar el tipo de notificaciones que se envían al microcontrolador en Modo Comando.

SET_PORTS_TO_OPEN: Instrucción del RBT-001 para indicar cuántos puertos físicos debe abrir el dispositivo.

SPP_ESTABLISH_LINK: Instrucción del RBT-001 para establecer un enlace de puerto serial con otro dispositivo Bluetooth. Necesita que se le indique la dirección del dispositivo remoto.

SPP_INCOMING_DATA: Indicación del RBT-001 al microcontrolador al recibir un dato por la conexión de puerto serial.

SPP_SEND_DATA: Instrucción para el envío de datos a través de una conexión de puerto serial en Modo Comando.

SPP_TRANSPARENT_MODE: Instrucción que configura el dispositivo RBT-001 en Modo Transparente luego de haberse establecido una conexión de puerto serial.

SPP: Serial Port Profile, perfil de la tecnología Bluetooh para simular puertos seriales RS-232.