UNIVERSIDAD AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

T E S I S

QUE PARA OBTENER EL TÍTULO DE
INGENIERO EN COMPUTACIÓN

TÍTULO:

"INTERFAZ DE MONITOREO PARA MÁQUINAS DE HEMODIÁLISIS FRESENIUS"

PRESENTAN:

MARICELA ANTONIO MÁRQUEZ
MARY CARMEN DE LA CRUZ CRUZ

DIRECTOR

ING. RUBÉN ANAYA GARCÍA

México D.F., Marzo 2010

AGRADECIMIENTOS

"A quienes me quieren o me han querido. A quienes me han hecho feliz o me han hecho llorar: ambos me han hecho sentir. Del sentimiento, yo he sacado conocimiento"

Susana Pérez-Alonso

A mi mamá, mi gran ejemplo. Gracias por tu eterna lucha para que nada falte en mi vida, por apoyarme en cada una de las cosas que hago, pero sobre todo gracias por siempre estar y brindarme tu amor. "Tú eres lo más bello que jamás me sucedió…TE AMO"

A mi papá, por brindarme tu amor, tu apoyo y, sobre todo, por hacerme más fuerte aunque eso haya significado tener días de guerras campales.

A ti, la persona más especial en mi vida, mi amiga, mi hermana, mi confidente. Gracias Pao por pedir hace tantos años a tres personas mágicas que yo viniera al mundo y sobre todo por estar presente cada vez que siento caer, gracias por quererme y soportarme tal y cómo soy.

A mis abuelitas (Q.E.P.D) quienes siento que me cuidan desde arriba. Siempre recuerdo su fortaleza, son mi inspiración para no desfallecer.

A mis amigos que me han dejado ser parte de su vida. Gracias por estar ahí para reír y llorar, siempre he sentido su apoyo y sus porras, los quiero mucho.

Al Ingeniero Ángel Hernández y al Ingeniero Rubén Anaya por brindarnos su conocimiento y apoyo para la culminación de este proyecto, así como un agradecimiento especial al M. en C. Víctor Manuel Bravo Rodríguez por su dirección y colaboración, gracias por compartirnos su experiencia y conocimiento.

Maricela A. M.

G*racias Dios por estar presente en mi vida y nunca dejarme sola, por ayudarme a cumplir mis metas y terminar mi trabajo pero sobre todo por permitirme llegar hasta este momento y poder compartirlo con las personas que más quiero.*

A mi madre quien con tanto esfuerzo y dedicación logro sacar a todos sus hijos adelante, por darme la mejor educación, por escucharme, comprenderme y soportarme durante esos momento de desesperación, por ser la mejor mamá y papá que dios me pudo haber dado. Te quiero mucho.

A mi hermana Leticia mil gracias por todo los cuidados, consejos sabios y enseñanzas, por ser una segunda madre cuando no pudo estar presente la nuestra, por guiarnos a mis hermanos y a mí por un buen camino y por todos los sacrificios que tuviste que hacer.

A mi hermano Alberto por enseñarme que siempre se puede salir a adelante no importando las circunstancias en las que te encuentres, por los cuidados y atenciones que me tuviste y principalmente por ser un padre para mí.

A mi hermana Lupita quien me enseño que en la vida se toman riesgos pero nunca hay que tener miedo, por demostrarme que la vida se disfruta y no se sufre.

A mi hermanito Miguel quien convivió todos los años de su vida conmigo por ser el menor y tener que soportarme tanto, gracias por tus cuidados, consejos y ocurrencias para hacerme reír cuando no tenía ganas, por enseñarme a valorar tanto la vida y todos los momentos mágicos que trae consigo.

A Víctor por estar presente en las etapas de transición más difíciles de mi vida y aguantar todo, por apoyarme tanto en este proyecto así como compartirme sus conocimientos, por ser tan sencillo, listo, caballeroso y gentil, pero sobre todo porque llegaste a mi vida cuando menos lo esperaba y llegaste para hacerme tanto bien.

Al mis asesores el Ingeniero Ángel por darme la confianza para llevar a cabo dicho proyecto y con esto ayudar a tanta gente, al Ingeniero Rubén Anaya por brindarnos sus enseñanzas y apoyarnos en la realización de dicho proyecto.

Mary Carmen De La Cruz

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La Medicina y en general las Ciencias de la Salud se han beneficiado de los avances tecnológicos, brindando mejores alternativas de diagnóstico y recuperación a los pacientes. Por su parte, el progreso en los sistemas informática y las aplicaciones multimedia han permitido mejorar procesos destinados al tratamiento, interpretación, almacenamiento, presentación y transmisión de información biomédica relacionada con el estado clínico o patológico de un paciente.

Dentro de sus aplicaciones se destacan el monitoreo remoto de variables fisiológicas, en el cual parámetros biomédicos específicos de un paciente son transmitidos desde la comodidad de su hogar o sitio de trabajo a una central de monitoreo, generándose alarmas en caso de detectarse algún parámetro por fuera de su rango permitido o en este caso para la pronta evaluación y diagnostico de una Sesión de Diálisis

En los centros clínicos y hospitalarios, la supervisión de pacientes ubicados en diversas áreas o unidades es de primordial importancia, debido a que estos son personas que pueden sufrir repentinos cambios en el comportamiento de sus signos vitales, razón por la cual se hace imprescindible un seguimiento continuo de sus señales biomédicas.

En el proyecto Implementación de una central de monitoreo y registro de parámetros fisiológicos para máquinas de Diálisis Fresenius se pretende que el personal de un hospital pueda hacer de una manera más eficiente el seguimiento clínico continuo de un paciente con insuficiencia renal a través de una aplicación que trasmite los parámetros fisiológicos a través de la red, lo que favorece la monitorización en tiempo real.

PLANTEAMIENTO DEL PROBLEMA

El Instituto de Ciencia Médicas y Nutrición cuenta con un área llamada Unidad Metabólica donde son atendidos personas que padecen de una enfermedad conocida como insuficiencia renal, dichos pacientes llevan un tratamiento a través de sesiones de Diálisis gracias a una máquina conocida como “Máquina de Hemodiálisis” donde se registran algunos de sus parámetros fisiológicos.

Actualmente el personal tiene que anotar cada cierto tiempo los parámetros fisiológicos de cada paciente por cada unidad de Diálisis, ya que no se cuenta con una central de monitoreo para este fin, lo que hace más lento el seguimiento de su diagnóstico.

OBJETIVO

Diseñar, construir, programar un sistema de monitoreo que sea capaz de obtener, mostrar y almacenar los parámetros fisiológicos de las unidades de hemodiálisis.

JUSTIFICACIÓN

Insuficiencia Renal Crónica (ICR) es la consecuencia de un daño prolongado e irreversible acompañado de numerosas y variables alteraciones metabólicas[1]

La ICR es una enfermedad degenerativa que afecta de forma importante la vida familiar, escolar, laboral y social del paciente y la gente que lo rodea. Estadísticas demuestran que dicha enfermedad ha ido en aumento constituyendo un problema de salud pública que genera un elevado costo social y económico; tanto de la propia enfermedad como de los tratamientos de sustitución.

En México la insuficiencia renal es una de las principales causas de atención hospitalaria, ocupando el 4º lugar en hombres y el 10º por las mujeres. En el 2001, el sector público reportó a la insuficiencia renal como el 9º lugar de mortalidad. Así mismo, en ese año, la OMS reportó una mortalidad mundial total de 625 000 casos para enfermedades renales como nefritis y nefrosis.

De acuerdo al censo del Instituto Nacional de Estadística y Geografía (INEGI) acerca del porcentaje de casos de morbilidad hospitalaria del 2001 al 2006. Las enfermedades del sistema urinario se encuentran entre las cinco principales causas de morbilidad por egreso hospitalario. **[1]**

Entidad federativa Principales causas de morbilidad	2001			2002			2003			2004			2005			2006		
	Total	Hombres	Mujeres	Total	Hombres	Mujeres	Total	Hombres	Mujeres	Total	Hombres	Mujeres	Total	Hombres	Mujeres	Total	Hombres	Mujeres
Estados Unidos Mexicanos																		
Embarazo, parto y puerperio	33.9	NA	49.4	33.9	NA	49.3	33.6	NA	48.9	33.5	NA	48.8	34.1	NA	49.4	34.4	NA	49.7
Traumatismos y envenenamientos	7.4	15.2	3.9	7.3	14.8	3.9	7.4	15.1	3.9	7.2	14.5	3.9	7.1	14.5	3.8	7.1	14.5	3.9
Enfermedades del sistema circulatorio	5.1	7.9	3.8	5.0	7.8	3.7	5.0	7.8	3.8	5.1	8.7	3.4	4.9	7.8	3.6	4.8	7.8	3.5
Enfermedades del sistema urinario	4.3	6.4	3.3	4.2	6.3	3.2	4.3	6.4	3.3	5.0	7.9	3.7	4.5	7.0	3.4	4.7	7.4	3.5
Ciertas afecciones originadas en el periodo perinatal	4.4	7.9	2.9	4.3	7.7	2.8	4.2	7.4	2.7	4.4	6.7	3.4	4.1	7.4	2.7	4.1	7.3	2.6

Tabla A. Porcentaje de casos de morbilidad hospitalaria por entidad federativa y principales causas según sexo, 2001 a 2006. INEGI

[1] García G. Insuficiencia Renal Crónica. Tratado Nefrología, Editor. Treviño B.A. México 2003 Tomo II pp 1152-1163

Actualmente en la Unidad Metabólica del Instituto Nacional de Ciencias Médicas y Nutrición Salvador Zubirán (INNSZ) se cuenta con el equipo necesario para elaborar el tratamiento requerido para aquellos pacientes que padecen insuficiencia renal.

El Instituto da servicio a los pacientes de insuficiencia renal toda la semana. La unidad metabólica cuenta con 12 máquinas de hemodiálisis de distintos modelos, cada una de ellas es ocupada por 3 ó 4 pacientes al día, los cuales reciben 3 sesiones de diálisis por semana.
Con los datos anteriores se calcula que en la unidad metabólica del INNSZ se realizan aproximadamente 84 sesiones al mes a 30 distintos pacientes.

En INNSZ, la recolección de datos fisiológicos del paciente en el momento que son sometidos al tratamiento de hemodiálisis se maneja de manera no automatizada y poco eficiente, debido a que el personal hospitalario obtiene esta información directamente de las máquinas de hemodiálisis, también llamadas riñón artificial, y es apuntada en hojas guardadas posteriormente en un archivo.

Este procedimiento hace lento la evaluación del paciente, además de que se necesita contar con un lugar en específico para archivar las hojas de los distintos pacientes, los cuales son sometidos al procedimiento de diálisis 4 horas 3 veces por semana.
Mediante el uso de una central de monitoreo se busca mejorar la eficiencia en el monitoreo y diagnóstico de los pacientes, a través de la automatización para obtener los parámetros fisiológicos mostrados en la máquina de hemodiálisis.

Los principales beneficios de contar con una central de monitoreo son:

- El personal puede abarcar a todos los pacientes, verificando los parámetros fisiológicos al mismo tiempo.
- No es necesario, al menos que sea caso de emergencia, que el personal hospitalario se encuentre dentro del área metabólica.
- En el momento de ser almacenados estos valores en una base de datos esta información puede ser consultada en cualquier momento.
- Rapidez en el seguimiento del diagnóstico.

CAPÍTULO I

ESTADO DEL ARTE

CAPÍTULO I ESTADO DEL ARTE

1.1 Estado del Arte

En este capítulo se abordarán en forma general los trabajos previos existentes al diseño de nuestro proyecto, así como los que existen en el mercado, lo que nos da una idea más amplia para después profundizar en el tema.

1.2 Central de Monitoreo Fresenius

La pantalla principal de la interfaz anteriormente mencionada es la siguiente:**[2]**

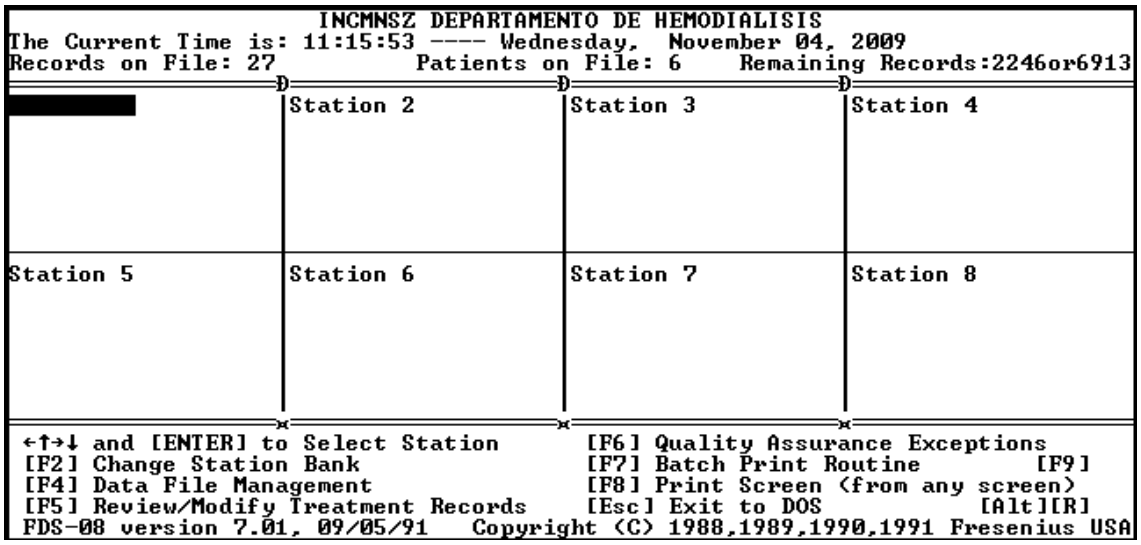


Figura 1.1 Menú principal Central de Monitoreo Fresenius

Con las flechas el usuario selecciona la estación a la cual se va a conectar algún nuevo paciente, al presionar Enter el usuario entra a la base de datos en la cual se puede seleccionar un paciente o en dado caso registrar uno nuevo.

A continuación observamos una pantalla con tres secciones diferentes. La primera sección contiene los datos del paciente (nombre, apellidos, número de registro, etc.), datos iníciales de la máquina (Baño K, Baño Ca, Baño dex, etc.) así como los parámetros de temperatura, presión peso, heparina, etc. pre y post diálisis.

En la segunda sección se observan los parámetros fisiológicos obtenidos de la máquina de hemodiálisis.

En la última sección se observa la fecha, el tiempo de conexión y un promedio de temperatura, el volumen total de sangre, etc.

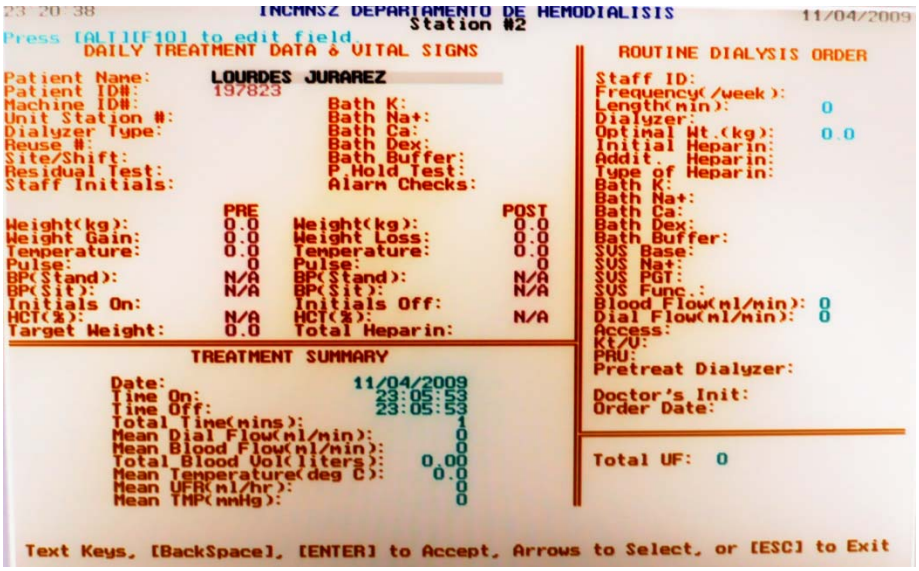


Figura 1.2 Seguimiento datos fisiológicos

Al salir de la pantalla anterior existe un menú con el cual podemos obtener un resumen de distintos datos del paciente, así como gráficas de los mismos datos, seguir observando los parámetros fisiológicos del paciente, incluir comentarios, revisar y/o introducir tratamiento médico del mismo. Todas las opciones anteriormente mencionadas se pueden imprimir, ya sea en pantalla o si se tiene una impresora conectada a la computadora.

Dentro del menú principal de la central de monitoreo se encuentra un menú por el cual también se pueden accesar a los datos históricos de los pacientes sin necesariamente estar conectados a la máquina de hemodiálisis por lo tanto también se puede observar los distintos datos que tuvieron en su sesión.

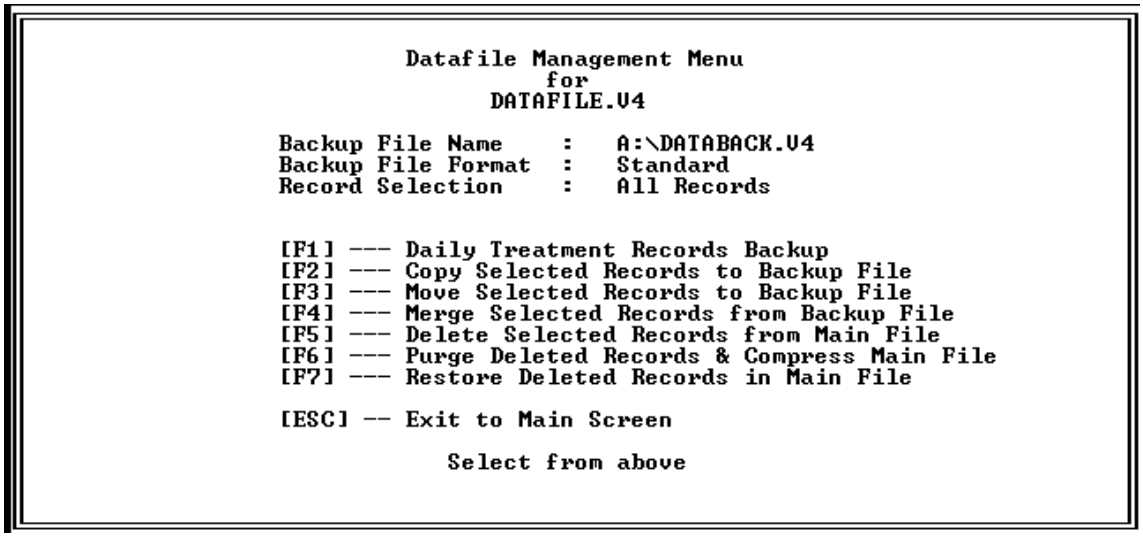


Figura 1.3 Menú datos históricos

1.3 Desarrollos previos.

1.3.1 Sistema de Telemonitoreo de Signos Vitales utilizando una Red LAN

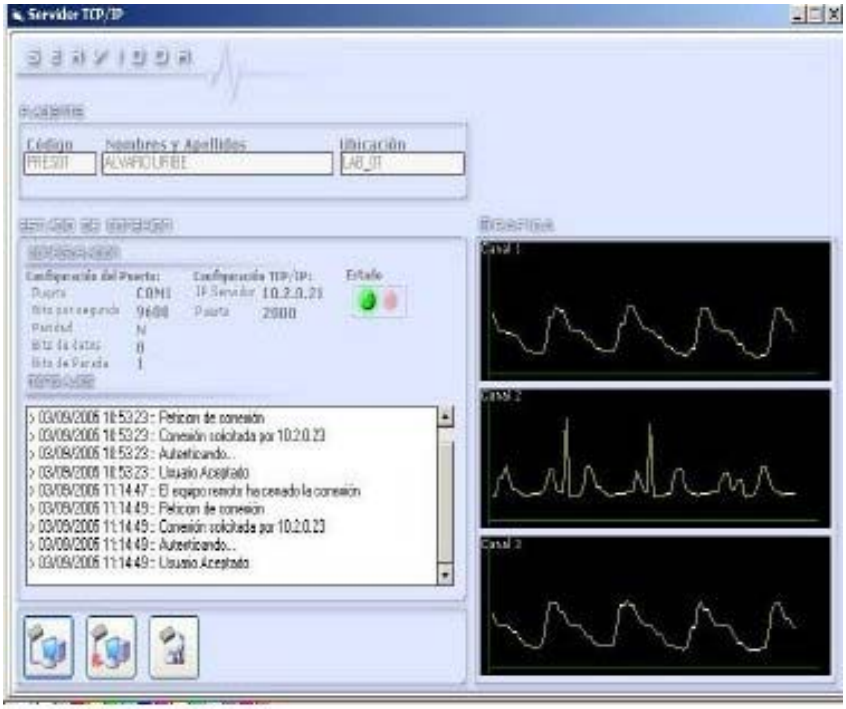


Figura 1.4 Sistema de telemonitoreo de Signos Vitales

Dicho trabajo fue desarrollado por el Centro de Bioingeniería Cebi-UAC y el Programa de Ingeniería Electrónica y Telecomunicaciones de la Universidad Autónoma del Caribe en la Ciudad de Barranquilla – Colombia, por Verónica Berrocal y Sue Gil, Elisa de la Ossa y Alejandro Romero Ingeniero Electrónico Especialista en Bioingeniería de la Universidad Distrital Francisco José de Caldas UDFJC de la ciudad de Bogotá. **[3]**

El proyecto consiste en transmitir a través de una red LAN tres variables fisiológicas: señal electro cardiográfica ECG, señal fotoplestismográfica y la forma de onda de actividad respiratoria, así como datos complementarios del paciente.

En su desarrollo utilizaron equipos de bioinstrumentación (simulador de señales ECG, modulo de adquisición y acondicionamiento para señal fotoplestismográfica medidor de frecuencia cardiaca, frecuencia respiratoria, temperatura). La información proveniente de los equipos es posteriormente convertida a un formato digital utilizando un microcontrolador PIC18F873A de microchip.

Posteriormente las señales son suministradas al PIC para su proceso de conversión, adicionalmente la trasmisión a la PC lo realizan por el USART del PIC y la trasmisión se realiza a través del puerto serial. La aplicación para el usuario final fue diseñada en el lenguaje Visual Basic.

Limitantes:

- Existen equipos ya muy comercializados y adquiridos por los hospitales que detectan esas señales biomédicas.
- Solo registran algunos parámetros fisiológicos y estos son para otras áreas del hospital.

1.3.2 Proyecto. Diseño y programación de una base de datos con parámetros fisiológicos para una central de máquina de hemodiálisis Fresenius 2008H

Este Proyecto de investigación fue realizado por Samuel Lara Navarrete como informe técnico de la opción curricular del Instituto Politécnico Nacional, Unidad Profesional Interdisciplinaria de Biotecnología (UPIBI) por título lleva el nombre de “Diseño y programación de una base de datos con parámetros fisiológicos para una central de máquina de hemodiálisis Fresenius 2008H **[4]**

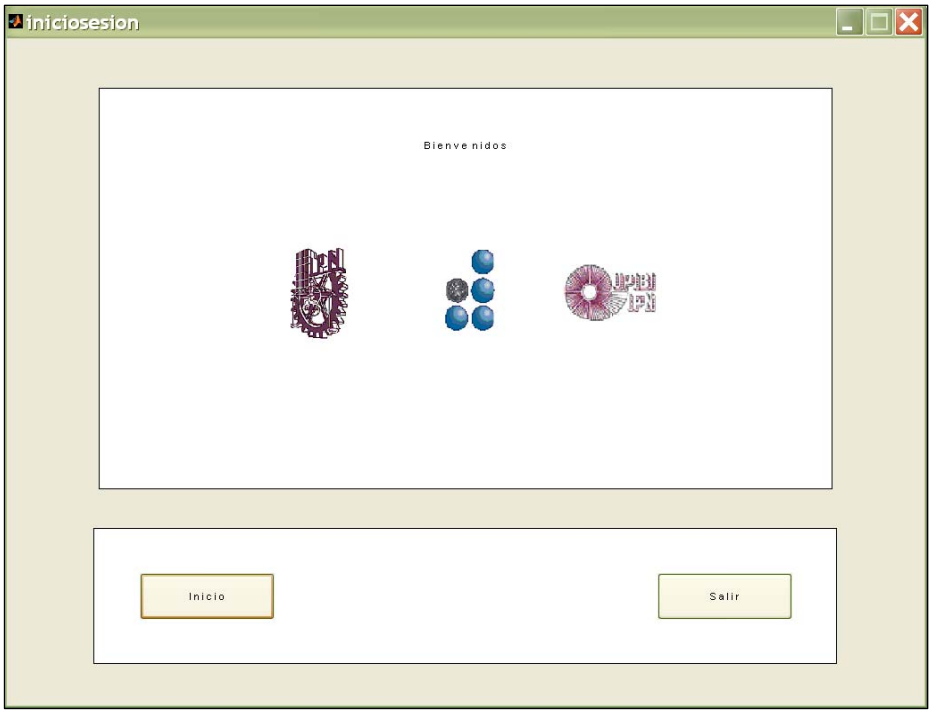


Figura 1.5 Inicio sesión

El desarrollo del proyecto consiste en la conexión de la computadora a la máquina de hemodiálisis por medio del puerto RS-232. Por medio de una función se le pide a la máquina cada determinado tiempo que arrojé los parámetros fisiológicos actuales.

La máquina de hemodiálisis arroja una cadena de caracteres y por medio de MATLAB se introduce estos valores a un arreglo y posteriormente separa la información de acuerdo al parámetro fisiológico que corresponde.

El sistema muestra una pantalla de inicio, en la cual se busca al paciente o se realiza su registro, dependiendo del caso.

Si es la primera sesión del paciente se piden datos personales para registrarlos en la base de datos, si no es la primera sesión se presenta otra pantalla en la cual se pide información importante antes de la diálisis del paciente, a continuación se selecciona la máquina la cual se conectará.

Durante el tiempo de la diálisis (3 a 4 horas) la computadora obtiene de la máquina los parámetros necesarios y los muestra en pantalla. Estos datos son mostrados en pantalla ser analizados y realizar un diagnóstico al paciente. Al terminar la diálisis en el sistema se guardan los parámetros fisiológicos finales del paciente.

En resumen el autor del proyecto consigue la comunicación entre la máquina de hemodiálisis y la computadora. La computadora muestra en pantalla los valores durante todo el procedimiento en el cual se encuentra conectado el paciente; sin embargo, estos valores no son guardados dentro de la base de datos, los únicos valores almacenados son los parámetros al inicio y al final de la sesión.
Se intentó realizar una gráfica para mostrar el estado físico del paciente en el transcurso de la diálisis sin embargo, este objetivo no fue logrado.

Actualmente este proyecto no se ha implementado dentro del hospital para el cual fue desarrollado.

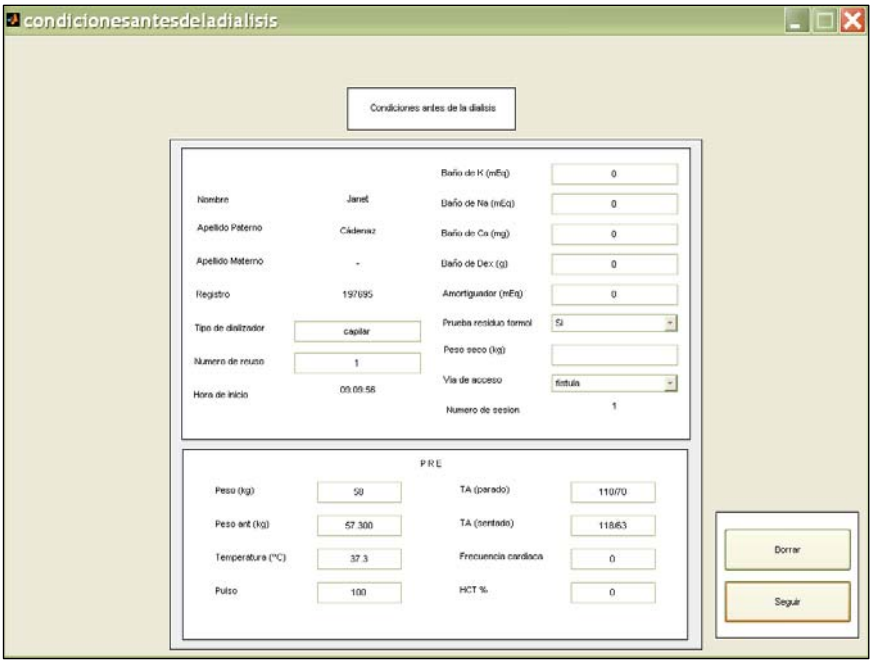


Figura 1.6 Datos paciente y datos prediálisis

Limitantes

- Se desarrolló sólo para un modelo de máquina en específico.
- La conexión sólo es lograda para una máquina, lo cual significa que se necesita una computadora por máquina de hemodiálisis.

1.3.3 Proyecto, “Desarrollo de una Interfaz de Usuario para una Máquina de Hemodiálisis”

Proyecto “Desarrollo de una Interfaz de Usuario para una Máquina de Hemodiálisis”, desarrollado por Olmos, K., Gómez, M., Rascon, L. para el Instituto de Ingeniería y Tecnología de la Universidad Autónoma de Ciudad de Juárez. **[5]**

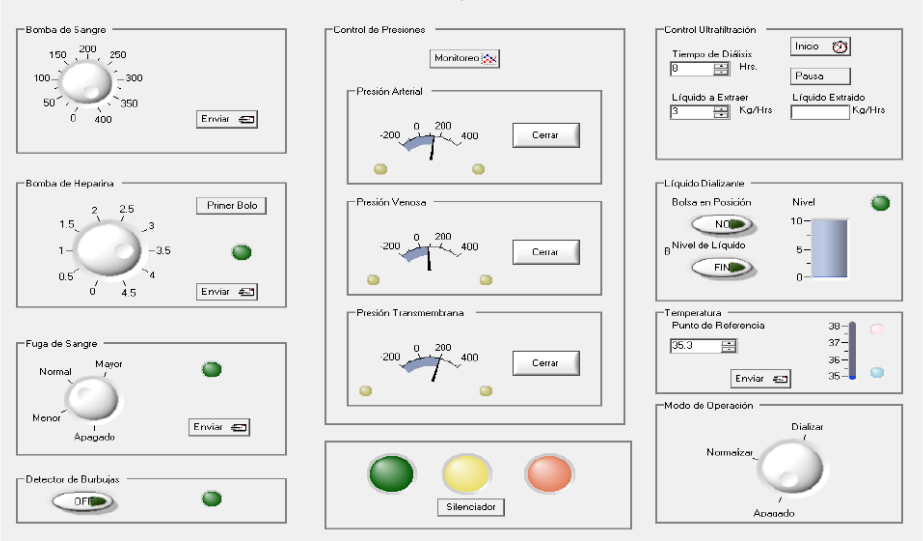


Figura 1.7 Pantalla Principal de Interfaz de Usuario para una máquina de hemodiálisis

La funcionalidad principal fue obtener una interfaz de usuario para la máquina de hemodiálisis Baxter 550.

El sistema de software permite configurar las especificaciones de la sesión de hemodiálisis y sirve como interfaz entre el operador y el sistema físico para realizar la sesión de diálisis. El sistema de control utilizado se dividió en dos secciones: una tarjeta maestra y cuatro tarjetas esclavas.

La tarjeta maestra realiza la comunicación entre la computadora y las tarjetas esclavas. Esta comunicación se lleva a cabo al recibir los códigos enviados por la computadora personal, analizarlos y reenviarlos a la tarjeta esclava correspondiente. El monitoreo y control del sistema físico se divide en cuatro secciones: bomba de sangre, bomba de heparina, control de temperatura y alarmas.

Se remplazó el sistema de control por una computadora personal que emula las funcionalidades del control. El software de emulación permite seleccionar los códigos que enviaría el sistema de control al comunicarse con la interfaz de usuario.

El funcionamiento detallado del software es el siguiente:

- El programa espera que el usuario introduzca sus datos y después de validarlos muestra el menú principal.
- Realizar sesión de hemodiálisis. En este proceso el programa envía un código de inicio al sistema de control y espera a que le regrese el código que representa el estado actual del sistema físico. Si el código es válido el operador debe elegir la temperatura a la cual se mantendrá el líquido dializante.
- Cuando se alcanza la temperatura establecida el sistema de software le indica al operador que puede introducir las especificaciones requeridas e inicia el tratamiento de diálisis.
- Si se registra alguna alarma, la misma aparece en la interfaz y si no es corregido termina la sesión. En caso contrario, al ser corregido, continua el procedimiento del programa
- Después de ese procedimiento, durante la sesión se muestran los parámetros fisiológicos de la máquina.

El programa se desarrolló utilizando el entorno de Visual Studio 2005 con el lenguaje de programación C# y utilizando los controles e indicadores proporcionados por Measurement Studio de National Instruments.

Durante el diseño, el sistema de control (hardware) es el responsable del sistema físico y la funcionalidad del sistema de software es servir de interfaz de usuario.

Limitantes:

- Se necesita una computadora por máquina.
- No se especifica si las lecturas mostradas en pantalla son almacenadas en una base de datos.

1.4 Métodos alternativos

A continuación se hace mención de las diferentes opciones para realizar el proyecto y sus limitantes, antes de elegir el método final, lo que ayudo a tener una mejor visión con respecto al desarrollo del proyecto.

1.4.1 Adaptador PCI-E de 8 puertos RS232

Utilizando una adaptador PCI-E de 8 puertos RS232, que se instalaría en la computadora principal de monitoreo, para poder así comunicar cada máquina de hemodiálisis directamente a la PC.

Esto se realizara mediante la modificación en el programa ya existente en Matlab (con la lectura de los puertos correspondientes).

Limitantes:

1. Distancia máxima permitida para este tipo de conexión.
2. Canalización distante a cables y fuentes de corriente.
3. Costo de la unidad $150 dólares más envió.

1.4.2 Convertidor USB a RS-232

Muchas computadoras modernas no incluyen puerto serie (RS232), ya que para aplicaciones informáticas se considera obsoleto. Sin embargo existen muchas aplicaciones en electrónica donde resulta muy conveniente usar el protocolo RS232 para el intercambio de información y la PC resulta la interfaz más conveniente.

En el mercado existen convertidores de USB a RS232 integrados en un cable o bien como adaptador. Lo que realizan estos adaptadores es emular un puerto serie mediante el puerto USB. Estos adaptadores vienen con un software que una vez instalado crea un puerto serie virtual a través del puerto USB.

Para realizar la comunicación por medio del RS232 los puertos van montados directamente a la computadora. Generalmente no existe ningún problema debido a que la aplicación se comunica directamente a través de un driver y el receptor asíncrono universal (UART). Por otro lado para un hub USB surge una congestión de comunicación.

Al conectarse con un USB se pueden generar varios problemas de congestión en la comunicación debido a la velocidad con la que trabajan estos dos puertos. Lo anterior tiene como resultado retrasos en la transmisión de la información.

Limitantes

- La conexión sólo es lograda para una máquina, lo cual significa que se necesita una computadora por máquina de hemodiálisis. Lo anterior genera un mayor gasto económico.

1.4.3 Unidades de Transmisión vía Bluetooth

Utilizar 8 unidades de transmisión de datos serie vía Bluetooth (Convertidores Bluetooth(TM) PSI-WL). Transmisión inalámbrica de las interfaces serie RS232, RS422 y RS485 de 2 hilos.

Posicionamiento óptimo de la antena mediante una conexión de antena externa, dependiendo de la concepción de la instalación, de esta manera se logra alcances de hasta 150 m (antena omnidireccional) o superiores (antena direccional).

Utiliza la banda ISM de 2,4 GHz y es un estándar industrial internacional, sin licencia y abierto.

Limitantes:

1. Máximo 7 máquinas conectadas a una central de monitoreo.
2. Interferencia que causa con otros equipos biomédicos.

CAPÍTULO II

MARCO TEÓRICO

CAPÍTULO II MARCO TEÓRICO

2.1 Fisiología del Riñón

Los riñones son los órganos que se encargan de filtrar el plasma y mantenerlo libre de los desechos nitrogenados, producto del metabolismo de las proteínas y de otras sustancias solubles en agua, así como mantener con medio químico con características ideales para la función celular.

Tienen forma de un fríjol y su longitud y peso varía desde 6cm y 24g, en un recién nacido, hasta 12 cm o más en un adulto. Se sitúan en la parte posterosuperior de la cavidad abdominal, uno a cada lado de la columna vertebral.

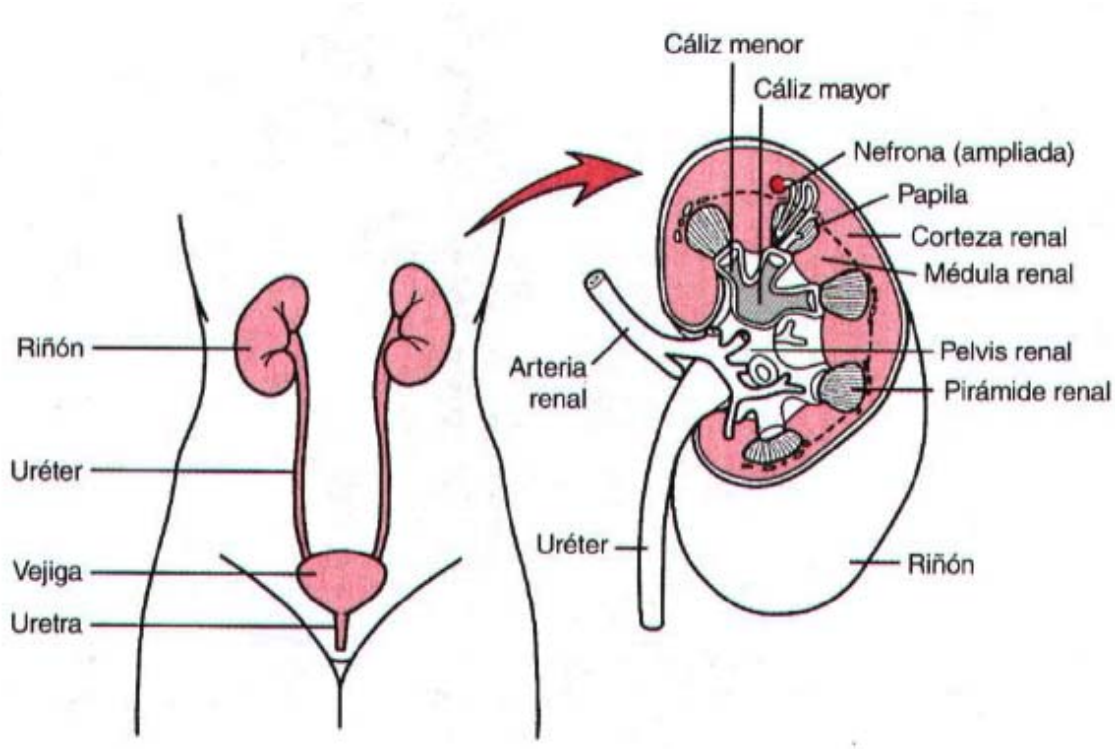


Figura 2.1 Organización general de los riñones y del sistema urinario ***[7]***

Dentro de los procesos renales básicos, se encuentra la formación de la orina, que empieza por la filtración del plasma esencialmente libre de proteínas, a través de los capilares glomerulares y túbulos renales, y es conducida a la pelvis renal por los túbulos colectores.

Los glomérulos funcionan como simples filtros a través de los que pasan el agua, las sales y los productos de desecho de la sangre, hacia los espacios de la cápsula de Bowman y desde allí hacía los túbulos renales. La mayor parte del agua y de las sales son reabsorbidas desde los túbulos y el resto es excretado como orina. La cantidad normal de orina eliminada en 24 horas de de 1.4 litros aproximadamente. **[6]**

La mayor parte de las personas suelen saber que los riñones tienen una función importante: eliminar del cuerpo sustancias de desecho que se han ingerido, como se menciono anteriormente.

Una segunda función, que es especialmente decisiva, es la regulación del volumen y la composición de los líquidos corporales. El equilibrio entre los ingresos (debidos a su ingestión o a su producción metabólica) y las pérdidas (debidas a la excreción o al consumo metabólico) se mantiene, en gran parte, por los riñones.

Esta función reguladora de los riñones mantiene el ambiente estable que todas la células necesitan para llevar a cabo sus diversas actividades.

Es importante mencionar que los riñones realizan numerosas funciones, como las siguientes:

- Excreción de los productos metabólicos de desecho y de las sustancias químicas extrañas (Los riñones eliminan también la mayoría de las toxinas y otras sustancias extrañas que se han producido por el cuerpo o han sido ingeridas, como los plaguicidas, los fármacos y los aditivos de los alimentos)
- Regulación del equilibrio hídrico y electrolítico (Para mantener la homeostasis, las excreción de agua y electrólitos de equiparse exactamente al ingreso de los mismos)
- Regulación de la osmolalidad de los líquidos corporales y de las concentraciones de electrólitos.
- Regulación de la presión arterial (Los riñones contribuyen a la regulación de la presión arterial a corto plazo mediante la secreción de factores o sustancias vasoactivas, como la retina, que da lugar a la formación de productos vasoactivos)
- Secreción, metabolismo y excreción de hormonas
- Glucogénesis (Los riñones sintetizan glucosa a partir de los aminoácidos y de otros precursores en situaciones de ayuno prolongado.

  En las enfermedades renales estas funciones desaparecen y rápidamente se producen graves alteraciones en los volúmenes y composición de los líquidos corporales) **[7]**

2.2 Insuficiencia Renal

Insuficiencia renal es la incapacidad de los riñones para mantener el plasma libre de impurezas, los electrolitos, así como el equilibrio interno del agua y del ácido base del organismo en su conjunto.**[8]**
Existen causas posibles de daño a los riñones, tales como:**[9]**

- Necrosis tubular aguda (NTA), causada por la falta de oxígeno en los tejidos renales o por la exposición a materiales que son tóxicos para el riñón.
- Enfermedad renal autoinmunitaria, como: síndrome nefrítico agudo o nefritis intersticial.
- Disminución del flujo sanguíneo debido a presión arterial muy baja, lo cual puede resultar de: quemaduras, deshidratación, hemorragia, lesión, cirugía.

- Trastornos que causan coagulación dentro de los vasos sanguíneos del riñón.
- Infecciones que causan lesión directamente al riñón.
- Complicaciones del embarazo, como: desprendimiento prematuro de placenta, placenta previa.
- Obstrucción de las vías urinarias

Síntomas.

- Heces con sangre
- Mal aliento
- Tendencia a la formación de hematomas
- Cambios en el estado mental o en el estado de ánimo
- Inapetencia
- Disminución en la sensibilidad, especialmente en las manos o en los pies
- Fatiga
- Dolor de costado (entre las costillas y las caderas)
- Temblor en la mano
- Hipertensión arterial
- Sabor metálico en la boca
- Náuseas o vómitos que pueden durar días
- Hemorragia nasal
- Hipo persistente
- Sangrado prolongado
- Crisis epiléptica
- Movimientos letárgicos y lentos
- Hinchazón generalizada por retención de líquidos
- Hinchazón de tobillos, pies y piernas
- Cambios en la micción
    - Disminución de la cantidad de orina
    - Micción excesiva durante la noche
    - Suspensión de la micción por completo

2.2.1 Diálisis

Nuestro organismo está compuesto por sustancias que cumplen leyes fisicoquímicas establecidas.

Es importante comprender los fenómenos que se producen en el cuerpo humano y también cuando son modificados o alterados por la enfermedad. En la atención al enfermo renal se intenta emular procesos biológicos.

La perdida grave de la función renal, ya sea aguda o crónica supone una amenaza para la vida y obliga a eliminar los productos tóxicos de desecho y a restablecer el volumen y composición de los líquidos orgánicos. Las opciones actuales de tratamiento renal sustitutivo incluyen la hemodiálisis, la diálisis peritoneal y el trasplante renal.

La hemodiálisis consiste en filtrar el exceso de líquidos y las sustancias tóxicas del organismo mediante el paso de la sangre del paciente por un filtro periódicamente.

La sangre del paciente se conduce entubada desde el organismo hasta la máquina de hemodiálisis (riñón artificial), está a su vez atraviesa un filtro de limpieza (dializador) en el que hay un intercambio entre la sustancia dializadora y la sangre, recogiendo todas las sustancias tóxicas de la sangre y así retornar de nuevo al cuerpo. Este tratamiento dura de 3 a 4 horas por sesión y se realizan 3 sesiones por semana. El tiempo necesario para cada sesión depende de:**[10]**

- El grado de funcionamiento de los riñones del paciente.
- La cantidad de líquido retenido entre una sesión y otra.
- Peso, estado físico y situación de salud del paciente.
- El tipo de riñón artificial que se utiliza.

Para poder realizar el proceso es necesario tener un acceso vascular, para la cual existen tres posibilidades:

- Realización de una fístula (unión de una vena y una arteria del antebrazo).
- Implantación de un injerto, el cual es un vaso artificial que une una vena con una arteria.
- Catéter externo

Procedimiento de la hemodiálisis **[11]**

1. La sangre con sustancias tóxicas sale del organismo a través de una aguja introducida en el acceso vascular e impulsado por una bomba, recorre un circuito extra-corporal, a través de un equipo arterial, y entra al dializador/filtro instalado en el equipo de hemodiálisis.
2. En la máquina, la sangre pasa por el dializador/filtro poniéndose en contacto con el baño de diálisis.
3. El baño de diálisis es una solución que, debido a su concentración y composición química, atrae a las impurezas y al agua contenida en la sangre. Las impurezas atraviesan la membrana y pasan al baño.
4. El baño de diálisis, que adquirió las impurezas es el agua de la sangre, sale de la máquina y se expulsan por el drenaje, siendo luego eliminadas al exterior de la máquina.

5. Ahora purificada, “limpia”, la sangre sale por el otro extremo de la máquina, volviendo al paciente a través del equipo y la aguja venosa.

Diálisis Peritoneal

La diálisis peritoneal se basa en el hecho fisiológico de que el peritoneo es una membrana vascularizada semipermeable, que mediante mecanismos de transporte osmótico y difusivo permite pasar agua y distintos solutos desde los capilares sanguíneos peritoneales al líquido dializado.
Existen tres tipos de diálisis peritoneal, los cuales son: **[12]**

- Diálisis Peritoneal Ambulatoria Continua (CAPD).

Se colocan litros de solución de diálisis en la cavidad abdominal y se dejan ahí de 4 a 5 horas. Posteriormente se drena dicho líquido y se desecha, colocando de nuevo solución, se practican 4 cambios al día.

- Diálisis Peritoneal Intermitente (CCPD)

Se lleva a cabo 3 veces por semana de 10 a 12 horas por tratamiento. En ocasiones se realiza una vez a la semana con 30 cambios.

- Diálisis Peritoneal Cíclica Continua (IPD)

Por lo general se practica en casa utilizando una máquina cicladora de diálisis peritoneal durante la noche, mientras el paciente duerme. Durante ese periodo la máquina realiza 4 a 6 cambios.

2.3 Máquina de Hemodiálisis

El riñón artificial es un aparato que elimina el exceso de iones y desechos que se acumulan en la sangre cuando se presenta una falla renal. La sangre es bombeada de una de las arterias del paciente a través de una serie de tubos bañados por una gran cantidad de líquido. Dicha tubería conduce luego la sangre nuevamente al paciente por medio de una vena. El líquido que se renueva constantemente es una solución de sal de concentraciones iónicas similares a las del plasma normal

El principio básico es simplemente al fluir la sangre a través de la tubería y las concentraciones de solutos tienden a equilibrarse en la sangre, de esta manera si la concentración plasmática de potasio se encuentra en el paciente por encima del nivel normal, el potasio se difunde saliendo de la sangre y entrando al liquido preparado y los excedentes se difunden a través de la tubería y quedan así eliminados del cuerpo.[2]

[2] Vander, Sherman y Luciano, Fisiología Humana, Riñón artificial, McGRAW-HILL, Segunda edición, México,1978 pp 276

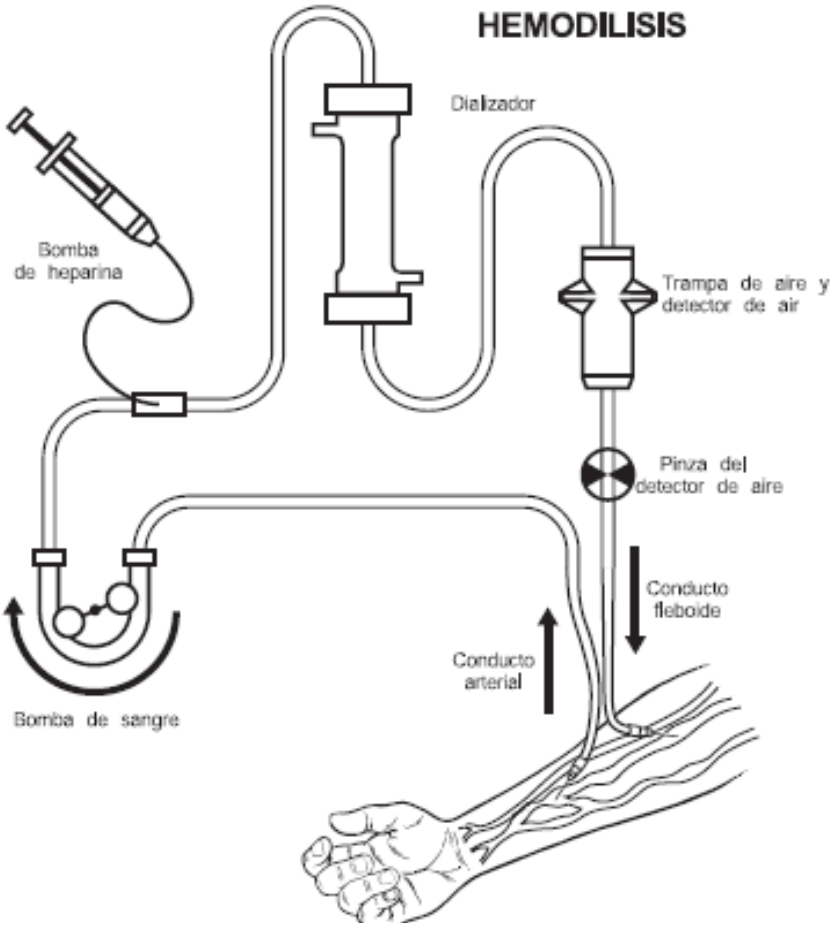


*Figura 2.2. Hemodiálisis, Formato informativo para pacientes renales **[13]***

El monitor de diálisis denominado riñón artificial es el aparato que junto con la instalación y material adecuado permite el proceso de la hemodiálisis, (diversas válvulas y alarmas), lo que asegura que el tratamiento está funcionando con seguridad, monitoreando la presión venosa, la velocidad con que su sangre circula por los tubos y la cantidad de agua que se elimina durante el tratamiento.**[13]**

2.4 Centrales de Monitoreo

El monitoreo permite caracterizar la evolución temporal de los parámetros de un sistema, inherentes al dominio de aplicación, dentro de la perspectiva de proveer un diagnóstico en caso de anomalías.[3]

[3] IV Latin American Congress on Biomedical Engineering 2007, Bioengineering Solutions for Latin America Health, Sistema Multiagentes para el Monitoreo Inteligente**,** Julio, Hernández, Beuchée, Wong, Passariello, Mora, Carrault, GBBA, Universidad Simon Bolivar, Caracas, Venezuela

Figura 2.3 Máquinas de Hemodiálisis en uso. Tomada el 18 de Abril del 2009

En el contexto médico, los sistemas de monitoreo están basados en la aplicación iterativa de cuatro etapas bien definidas: adquisición de datos, detección, diagnóstico y terapia.

Figura 2.4 Etapas del sistema de moniteo en el ambito hospitalario. Tomado de Sistemas Multiagentes para el monitoreo inteligente

2.5 Elementos Físicos de Diseño

2.5.1 Microcontroladores

Un microcontrolador es una computadora con CPU, memoria, entrada y salida, Puertos (I/O) integrados en un solo circuito chip. [4]

[4] Cady Fredrick M, Microcontrollers and Microcomputers , Oxford University Press, Primera Edición, NY, 1997 pp 2

Los microcontroladores como todo sistema de cómputo se dividen en las dos grandes arquitecturas:

La arquitectura Von Neumann

La arquitectura propuesta por John Von Neumann, muestra como la unidad central de proceso o CPU, está conectada a una memoria única que contiene las instrucciones del programa y los datos. Las dos principales limitaciones de esta arquitectura tradicional son: **[14]**

- La longitud de las instrucciones está limitada por la unidad de longitud de los datos, por lo tanto el microprocesador debe hacer varios accesos a memoria para buscar instrucciones complejas,

- La velocidad de operación (o ancho de banda de operación) está limitada por el efecto de cuello de botella que significa un bus único para datos e instrucciones que impide superponer ambos tiempos de acceso.

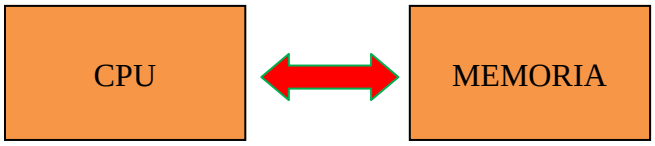


Figura 2.5 Arquitectura Von Neuman

La arquitectura Harvard

La arquitectura conocida como Harvard, consiste simplemente en un esquema en el que el CPU está conectado a dos memorias por intermedio de dos buses separados. Una de las memorias contiene solamente las instrucciones del programa, y es llamada Memoria de Programa. Las principales ventajas de esta arquitectura son: **[14]**

a) El tamaño de las instrucciones no está relacionado con el de los datos, y por lo tanto puede ser optimizado para que cualquier instrucción ocupe una sola posición de memoria de programa, logrando así mayor velocidad y menor longitud de programa,

b) El tiempo de acceso a las instrucciones puede superponerse con el de los datos, logrando una mayor velocidad de operación.

Los microcontroladores son computadores digitales integrados en un chip que cuentan con un microprocesador o unidad de procesamiento central (CPU), una memoria para almacenar el programa, una memoria para almacenar datos y puertos de entrada salida. A diferencia de los microprocesadores

de propósito general, como los que se usan en los computadores PC, los microcontroladores son unidades autosuficientes.

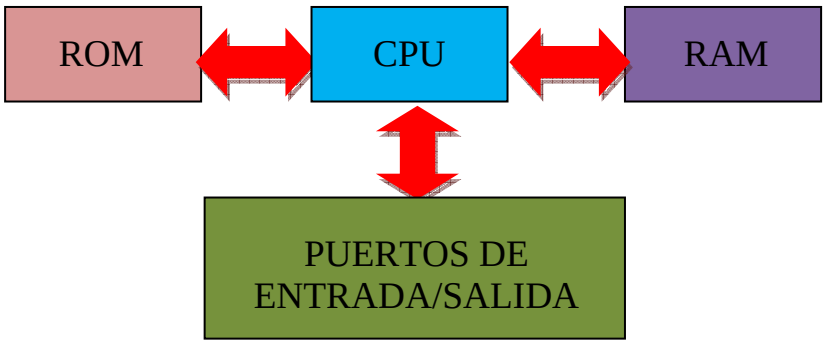


Figura 2.6 Arquitectura Harvard

El funcionamiento de los microcontroladores está determinado por el programa almacenado en su memoria. Este puede escribirse en distintos leguajes de programación. Además, la mayoría de los microcontroladores actuales pueden reprogramarse repetidas veces. **[14]**

2.5.1.1 Características de los microcontroladores.

Las principales características de los Microcontroladores son:

- Unidad de Procesamiento Central (CPU): Típicamente de 8 bits, pero también las hay de 4, 32 y hasta 64 bits con *arquitectura Harvard*, con memoria/bus de datos separada de la memoria/bus de instrucciones de programa, o *arquitectura de von Neumann*, también llamada *arquitectura Princeton*, con memoria/bus de datos y memoria/bus de programa compartidas.
- Memoria de Programa: Es una memoria ROM (*Read-Only Memory*), EPROM (*Electrically Programable ROM*), EEPROM (*Electrically Erasable/Programable ROM*) o Flash que almacena el código del programa que típicamente puede ser de 1 kilobyte a varios megabytes.
- Memoria de Datos: Es una memoria RAM (*Random Access Memory*) que típicamente puede ser de 1, 2 4, 8, 16, 32 kilobytes.
- Generador del Reloj: Usualmente un cristal de cuarzo de frecuencias que genera una señal oscilatoria de entre 1 a 40 MHz, o también resonadores o circuitos RC.
- Interfaz de Entrada/Salida: Puertos paralelos, seriales (UARTs, *Universal Asynchronous. Receiver/Transmitter*), I2C (*Inter-Integrated Circuit*), Interfaces de Periféricos

2.5.1.2 Familias y Marcas

Fabricante	Fabricante	Memoria Flash (Kb)	SRAM/ RAM	PORT I/O	USART/ UART	Puertos serie sincrono	I2C	Ethernet
Philips Simiconductor NPX	LCP2364 LCP2366 LCP2368	512	32/4	70	__/4	SI (2)	3	10/100
ATMEL	AT91SAM7X	256	64/	32	2/--	SI	NO	10/100
Microchip	18F97J60	128-256	/3808 bytes	70	2/--	SI (4)	SI	10/100
ST Microelectronics	STM32F107Rx STM32F107Vx	256	64/64	80	5/__	SI	2	10/100
Freescale	MCFS2259	512	64/	56	__/3	SI	2	10/100

Fabricante	Interfaz MAC/SPI	Oscilador Interno (MHz)	PWM	Empaquetado (pines)	Convertidor AD/DA	Precio por unidad (dolares)
Philips Simiconductor NPX	SI	4	SI	100	ADC	8.7
ATMEL	SI	18.43	SI (4)	100	ADC	20
Microchip	SI	41.66	SI (4)	100	ADC	4.9
ST Microelectronics	SI	25	SI (1)	100	ADC/DAC	12.44
Freescale	SI	48	SI (8)	100	ADC	7.5

Tabla 2.1 Comparación entre microcontroladores

2.5.1.3 Formas de Programación del Microcontrolador

Existen diversas formas de programar un PIC, usando diferentes lenguajes como son C, ó lenguaje Ensamblador. Así también existen muchos proveedores de software especializados en ofrecer compiladores, algunos de ellos ofrecen versiones gratis o "Demos". A continuación se describe cada uno de ellos:

Lenguaje Ensamblador:

El lenguaje Ensamblador es un tipo de lenguaje de bajo nivel utilizado para escribir programas y constituye la representación más directa del código máquina específico para cada arquitectura de computadoras.

Algunos dispositivos programables (como los microcontroladores) aun cuentan con esté lenguaje como única manera de ser manipulados.

La tarea fundamental de un ensamblador es traducir un programa en lenguaje de ensamblador al código correspondiente en lenguaje máquina. Es la primera abstracción del Lenguaje de Máquina, consiste en asociar a los códigos de operación palabras clave que faciliten su uso por parte del programador. **[15]**

Características:

- El lenguaje ensamblador es difícilmente portable, es decir, un código escrito para un microprocesador en particular necesita ser modificado muchas veces en su totalidad para poder ser usado en otro microprocesador.
- Al programar cuidadosamente en Lenguaje ensamblador se puede crear programas que se ejecutan más rápidamente y ocupan menos espacio.
- Cada una de sus sentencias es una codificación simbólica de una instrucción numérica máquina.

Compilador CCS

Ya después de conocer el lenguaje ensamblador es muy útil aprender a programar con un lenguaje de alto nivel como C. El compilador CCS C permite desarrollar programas en C enfocado a PIC con la ventaja que supone tener un lenguaje desarrollado específicamente para un microcontrolador concreto.

El Compilador C de CCS dispone de una amplia librerías de funciones predefinidas, comando de pre-procesado, además suministra los controladores para diversos dispositivos como LCD, convertidores AD, relojes de tiempo real, EEPROM serie, etc.

Un compilador convierte el lenguaje de alto nivel a instrucciones en código máquina; un *cross-compiler* Es un compilador que funciona en un procesador normalmente en una PC. El compilador CCS es un *cross-compiler.* **[16]**

Compilador C18

El lenguaje C fue creado en los años 70 para escribir el código del sistema operativo UNIX. Tanto por su origen como por sus características, es un lenguaje muy adecuado para la programación de sistemas ya que combina la abstracción de los lenguajes de alto nivel con la eficiencia del lenguaje máquina.

En 1983 se creó el comité técnico X3J11 para proporcionar una definición de lenguaje clara e independiente y en 1989, el estándar fue aprobado. ANSI Cooperó con ISO (Organización Internacional para Estandarización) para estandarizar C a nivel mundial.

Entre las características de este lenguaje cabe citar que es altamente portable, es muy flexible, genera código eficiente y permite escribir código muy compacto. **[17]**

El compilador MPLAB C18 es un compilador que optimiza el estándar ANSI C en los microcontroladores PIC18.

Y tiene las siguientes características:

- Compilador cruzado de lenguaje C para la serie de microcontroladores Microchip PIC 18
- Sigue la norma ANSI C
- Se integra en el entrono MPLAB IDE
- Números reales float y double de 32 bits.
- Gran variedad de librerías.
- Acceso transparente en la lectura /escritura de la memoria.
- Versión estudiantil gratuita.

2.6 Protocolos de Comunicaciones

Un protocolo de comunicaciones es un conjunto de normas que están obligadas a cumplir todas las máquinas y programas que interviene en una comunicación de datos entre ordenadores sin las cuales la comunicación resultaría imposible.

2.6.1 Comunicaciones punto a punto

Son los protocolos más antiguos y elementales utilizados para la comunicación mediante una línea de datos entre dos únicos ordenadores. Alguna de sus normas básicas establecen os siguientes criterios.

- El papel que asume cada una de las dos partes durante una sesión de comunicaciones, identificándose y definiendo el papel correspondiente al ordenador que ha iniciado la sesión y al que responde.
- Manera de controlar la correcta recepción de los datos.
- Tiempo máximo que debe pasar entre el envío de un mensaje y la recepción del acuse de recibo desde la estación receptora.

2.6.1.1 Serial

El estándar estipula las características mecánicas, eléctricas, y de un protocolo orientado al enlace físico punto a punto. Las características eléctricas que ofrecen una alta inmunidad a ruidos y distancia de enlace de hasta 16 metros dependiendo de la velocidad de trasmisión utilizada.**[18]**

La comunicación Serial consiste en trasmitir los datos a través de una única línea de datos con dependencia de formato. Y el usar este tipo de comunicación como resultado

- Menor disposición a errores
- Número reducido de líneas
- Serialización/Deserialización de datos
- Protocolo de Comunicación

Tipos de Comunicaciones Seriales

- Simplex: Transmisión en un solo sentido
- Half dúplex: Transmisión en ambos sentidos per no simultáneamente
- Full dúplex: Trasmisión en ambos sentidos simultáneamente

Existen dos tipos de Comunicación: Síncrono y Asíncrono

La transmisión Síncrona permite mayores velocidades de trasmisión mientras la transmisión Asíncrona mayor variabilidad de dispositivos a interconectar, a continuación se detalla mejor.

Serie síncrona

- Los datos son enviados en bloques el transmisor y el receptor son sincronizados por uno o mas caracteres especiales llamados caracteres sync.

Serie asíncrona

- Existe una línea de datos y una línea de tierra común a los dos comunicantes.
- En una transmisión asíncrona, un bit identifica su bit de comienzo y 1 o 2 bits identifican su final, no es necesario ningún carácter de sincronismo.
- Los bits de datos son enviados al receptor después del bit de start. El bit de menos peso es transmitido primero
- Un carácter de dato suele consistir en 7 o 8 bits.

- Dependiendo de la configuración de la transmisión un bit de paridad es enviado después de cada bit de datos y se utiliza para el control de errores en los caracteres de datos

- Finalmente 1 o 2 bits de stop son enviados.

Las señales más utilizadas se listan a continuación:
DTR (Data-Terminal-Ready): El PC indica al modem que esta encendido y listo para enviar datos
DSR (Data-Ser-Ready): El modem indica al PC que esta encendido y listo para transmitir o recibir datos.
RTS (Request-To-Send): El PC pone esta señal a 1 cuando tiene un carácter listo para ser envidado
CD (Carrier-Detect): El modem pone eta señal a 2 cuando ha detectado el ordenador.
CTS (Clear-To-Send): El modem está preparado para transmitir datos. El ordenador empezara a enviar datos al modem.
TxD: El modem recibe datos desde el PC.
RxD: El modem transmite datos al PC.

2.6.1.2 Paralelo

En 1981 IBM (International Business Machines) introdujo la Computadora Personal. El puerto paralelo (Standart Parallel Port o SPP) estaba incluido en el primer PC y se agregó a este como una alternativa al bajo rendimiento del puerto serial, para utilizarlo como controlador de las impresoras de matriz de punto de alto desempeño. **[19]**

Este puerto tenía la capacidad de trasmitir 8 bits de datos a la vez (del PC a la impresora), mientras que el puerto serial lo hacia uno en uno. En el momento que el puerto paralelo fue presentado, las impresoras de punto fueron el principal dispositivo externo que se conecto a éste.

En la actualidad se conoce cuatro tipos de puerto paralelo:

- Puerto paralelo estándar (Standart Parallel Port SPP)

- Puerto paralelo PS/2 (bidireccional)

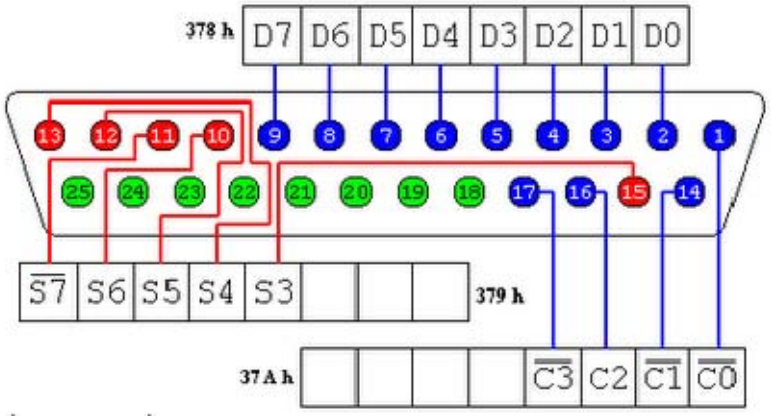
- Enhanced Parallel Port (EPP)

- Extended Capability Port (ECP)

Cada adaptador de puerto paralelo tiene tres direcciones sucesivas que se corresponde con otros tanto registros que sirven para controlar el dispositivo. Son el registro de salida de datos, el registro de estado y el registro de control.

El registro de salida es la dirección en que hay que poner cualquier carácter que sea dirigido al puerto (generalmente una impresora); el de estado contiene información sobre el dispositivo conectado, en especial la ocurrencia de posibles errores. El registro de control permite inicializar el puerto y controlar la transferencia.

El conector normalmente empleado en esta interface un conector macho DB-25, ver figura. Esta interfaz es rápida y generalmente su uso se reserva para las impresoras en lugar de la intercomunicación entre computadoras. El único problema con el puerto paralelo es que los cables no se deben extender para una gran longitud sin la amplificación de la señal, porque ocurrirían errores en los datos al momento de transmitir.



Pin	DESCRIPCIÓN	I / O
1	- Strobe	Out
2	+ Data Bit 0	In/Out
3	+ Data Bit 1	In/Out
4	+ Data Bit 2	In/Out
5	+ Data Bit 3	In/Out
6	+ Data Bit 4	In/Out
7	+ Data Bit 5	In/Out
8	+ Data Bit 6	In/Out
9	+ Data Bit 7	In/Out
10	- Acknowledge	In
11	+ Busy	In
12	+ Paper End	In
13	+ Select	In
14	- Auto Feed	Out
15	- Error	In
16	- Initialize Printer	Out
17	- Select Input	Out
18	- Data Bit 0 Return (GND)	
19	- Data Bit 1 Return (GND)	
20	- Data Bit 2 Return (GND)	
21	- Data Bit 3 Return (GND)	
22	- Data Bit 4 Return (GND)	
23	- Data Bit 5 Return (GND)	
24	- Data Bit 6 Return (GND)	
25	- Data Bit 7 Return (GND)	

Figura 2.7 Vista física de un conector puerto paralelo de una PC compatible

2.6.1.3 USB

El USB o Universal Serial Bus es una interfaz para la transmisión serie de datos desarrollado por empresas líderes del sector de las telecomunicaciones y de los ordenadores y que ha sido introducida en el mercado de los PC´s y periféricos para mejorar las lentas interfaces serie (RS-232) y paralelo. Provee una mayor velocidad de transferencia (de hasta 100 veces más rápido) comparado con el puerto Paralelo de 25-pin y el Serial DB-9, DB-25, RS-232 que son los puertos que se encuentran en la mayoría de los computadores.

En principio, la tecnología de bus serial universal (Universal Serial Bus o USB) comenzó a aparecer en computadoras personales (PC) en 1995.

Esta interfaz de 4 hilos distribuye 5V para la alimentación y puede transmitir datos a una velocidad de hasta 480 Mbps en su versión 2.0. Es un bus serie que hace posible la conexión de hasta 127 periféricos a una única puerta de un PC, con detección y configuración automáticas, siendo esto posible con el PC conectado a la red y sin tener que instalar software adicional, y sin tener que reiniciar el ordenador (plug and play, algo que con los puertos convencionales serie y paralelo no sucedía.

Características generales del USB **[20]**

La especificación del USB proporciona una serie de características que pueden ser distribuidas en categorías. Estas características son comunes para todas las versiones (desde la 1.0 hasta la 2.0)

Fácil uso para los usuarios:

- Modelo simple para el cableado y los conectores.
- Periféricos auto-identificativos.

Flexibilidad

- Amplio rango de tamaños de paquetes, permitiendo variedad de opciones de buffering de dispositivos.
- Gran variedad de tasas de datos de dispositivos acomodando el tamaño de buffer para los paquetes.

Amplia gama de aplicaciones y cargas de trabajo

- Adecuando el ancho de banda desde unos pocos kbs hasta varios Mbs.
- Soporta tanto el tipo de transferencia isócrono como el asíncrono sobre el mismo conjunto de cables.
- Soporta hasta 127 dispositivos físicos.
- Soporta la transferencia de múltiples datos y flujos de mensajes entre el host y los dispositivos.

Robustez

- Manejo de errores y mecanismos de recuperación ante fallos implementados en el protocolo.
- Inserción dinámica de dispositivos.

2.6.2 Comunicación en Red

Se establece a través de un protocolo, el cual es un método establecido de intercambiar datos en Internet. Un protocolo es un método por el cual dos ordenadores acuerdan comunicarse, una especificación que describe cómo los ordenadores hablan el uno al otro en una red.

El protocolo determina lo siguiente:

- El tipo de comprobación de errores que se utilizará.
- El método de compresión de los datos, si lo hay.
- Cómo indicará el dispositivo que envía que ha acabado el enviar un mensaje.
- Cómo indicará el dispositivo que recibe que ha recibido un mensaje.

2.6.2.1 Ethernet

Es un estándar de transmisión de datos para redes de área local y es probablemente el estándar más popular. A fines de 1996 más del 80% de las redes instaladas en el mundo eran Ethernet.
Fue desarrollado inicialmente en 1973 por el Dr. Robert M. Metcalfe en el compañía Xerox, como un sistema de red denominado Ethernet Experimental. El objetivo era conseguir un medio de comunicación entre computadoras.

Estos primeros trabajos contribuyeron substancialmente a la definición de la norma IEEE 802.3, que define el método de acceso CSMA/CD (Acceso Múltiple por Detección de Portadora con Detección de Colisiones). Esté se basa en que cuando un equipo DTE (“Data Terminal Equipment”) conectado a una LAN (Red de área local) desea transmitir, se mantiene a la escucha hasta que ningún equipo está transmitiendo, una vez que la red está en silencio, el equipo envía el primer paquete de información.

Los estándares Ethernet no necesitan especificar todos los aspectos y funciones necesarios en un Sistema Operativo de Red.

La especificación Ethernet se refiere solamente a las dos primeras capas del modelo OSI:

- La capa física: el cableado y las interfaces físicas.
- La de enlace: que proporciona direccionamiento local, detección de errores, y controla el acceso a la capa física. **[21]**

IEEE:

Corresponde a las siglas de *The Institute of Electrical and Electronics Engineers*, el Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial fundada en 1884 y dedicada a la estandarización.

NORMA IEEE 802

IEEE 802 es un comité y grupo de estudio de estándares perteneciente al Instituto de Ingenieros Eléctricos y Electrónicos que actúa sobre Redes de Ordenadores, concretamente y según su propia definición sobre redes de área local (LAN) y redes de área metropolitana (MAN)

La IEEE 802 es el nombre de un comité de estandarización del IEEE y por su extensión se denomina así los estándares por él producidos.

La primera versión fue un intento de estandarizar Ethernet aunque hubo un campo de la cabecera que se definió de forma diferente. Posteriormente ha habido aplicaciones sucesivas al estándar que cubrieron las aplicaciones de velocidad, redes virtuales, hubs, conmutadores y distintos tipos de medios, tanto de fibra óptica como de cables de cobre (tanto de par trenzado como coaxial).
Los estándares de este grupo no reflejan necesariamente o que usa en la práctica aunque a diferencia de otros grupos este suele estar cerca de la realidad.

2.6.2.2 Token-Ring

Este es el término utilizado para referirse a la norma IEEE 802.5 para implementar una red LAN. Tecnología creada originalmente por IBM en los años 70 y la segunda en popularidad, después de Ethernet. **[22]**

Funcionamiento:

- La red Token Ring consta de un conjunto de nodos conectados en forma de anillo.
- Los datos siempre fluyen en la misma dirección.
- Cada nodo recibe frames del nodo que le antecede y envía frames al nodo que le sigue.
- El anillo es un medio compartido: sólo un nodo (aquel que posee el token) transmite frames durante cierto tiempo.

Los datos en Token-Ring se transmiten a 4 ó 16 Mbps, depende de la implementación que se haga. Todas las estaciones se deben de configurar con la misma velocidad para que funcione la red. Cada computadora se conecta a través de cable Par Trenzado ya sea blindado o no, aun concentrador, y aunque la red quede físicamente en forma de estrella, lógicamente funciona en forma de anillo por cual da vueltas.

2.6.2.3 Wi-Fi

La norma IEEE 802.11 estableció en junio de 1997 el estándar para redes inalámbricas. Debido a la necesidad de los usuarios de redes LAN (Redes de Área Local) por tener movilidad y eliminar las desventajas que provoca una conexión física cableada a la red, surge la idea de crear redes inalámbricas de computadoras que ofrecieran los mismo beneficios de las redes LAN pero eliminando la conexiones cableadas entre nodos.

La especificación IEEE 802.11 es un estándar internacional que define las características de una red de área local inalámbrica (WLAN). Wi-Fi (que significa "Fidelidad inalámbrica") es el nombre de la certificación otorgada por la Wi-Fi Alliance, grupo que garantiza la compatibilidad entre dispositivos que utilizan el estándar. Una red Wi-Fi es en realidad una red que cumple con el estándar 802.11. **[23]**
Una red de área local inalámbrica puede definirse como "*Una red de alcance local que tiene como medio de transmisión el aire*".

Una red de área local inalámbrica o WLAN utiliza ondas electromagnéticas (radio e infrarrojo) para enlazar (mediante un adaptador) los equipos conectados a la red, en lugar de los cables coaxiales o de la fibra óptica que se utilizan en las LAN (redes de área local) convencionales cableadas (Ethernet, Token-Ring, etc.)

Topologías WLAN **[24]**

- Topología *peer to peer*: El cliente dentro de una célula se comunica directamente con otros
- Topología basada en puntos de acceso: usa a estos mismo como puente entre el tráfico en una red cableada (Ethernet)
- Topología de puente punto a multipunto: Un ejemplo es, cuando los puentes inalámbricos conectan una LAN en un edifico a otro LAN que se encuentra en otro edificio.

Sin embargo la necesidad de mayores velocidades de transmisión y mecanismos de seguridad más eficientes, han motivado el desarrollo de nuevos estándares para redes inalámbricas, tomando como base el estándar 802.11. Estos estándares se presentan a continuación

- IEEE 802.11b: Surge por la necesidad de incrementar la velocidad de transmisión, el cual tiene una velocidad máxima de transmisión de 11Mbps y trabaja en la frecuencia de 2.4 GHz.
- IEEE 802.11a: Opera en una banda de 5 GHz y su velocidad máxima de transmisión es de 54Mbps.
- IEEE 802.11g: Utiliza la banda de 2.4 GHz pero opera a una velocidad teórica máxima de 54 Mbps, o cerca de 24.7 Mbps de velocidad real de transferencia.

2.7 Interfaz de Comunicación Serial

El concepto de comunicación serial es sencillo. El puerto serial envía y recibe bytes de información un bit a la vez. Aun y cuando se dice que es un método más lento que el paralelo, este es más sencillo y puede alcanzar mayores distancias, utilizando comunicación serial el largo del cable puede llegar a los 1200 metros.

Típicamente, la comunicación serial se utiliza para transmitir datos en formato ASCII. Para realizar la comunicación se utilizan 3 líneas de transmisión:

- Tierra (o referencia)
- Transmitir
- Recibir.

2.7.1 Protocolo Serial

La comunicación serial es un protocolo muy común para comunicación entre dispositivos y también muy utilizado por varios dispositivos para instrumentación. Además, la comunicación serial puede ser utilizada para adquisición de datos si se usa en conjunto con algún dispositivo. **[25]**

Antes de iniciar cualquier comunicación con el puerto RS.232 se debe de determinar el protocolo a seguir dado que el estándar del protocolo no permite indicar en qué modo se está trabajando, es la persona que utiliza el protocolo el que debe decir y configurar ambas partes antes de iniciar la transmisión de datos.

Para que dos puertos se puedan comunicar, es necesario que las características sean iguales, siendo los parámetros a configurar los siguientes:

a. **Velocidad de transmisión *(baud rate)*:** Indica el número de bits por segundo que se transfieren, y se mide en baudios (*bauds*).

   Por ejemplo, 300 baudios representa 300 bits por segundo. Cuando se hace referencia a los ciclos de reloj se está hablando de la velocidad de transmisión.

b. **Bits de datos:** Se refiere a la cantidad de bits en la transmisión. Cuando la computadora envía un paquete de información, el tamaño de ese paquete no necesariamente será de 8 bits. Las cantidades más comunes de bits por paquete son 5, 7 y 8 bits. El número de bits que se envía depende en el tipo de información que se transfiere.

Por ejemplo, el ASCII estándar tiene un rango de 0 a 127, es decir, utiliza 7 bits; para ASCII extendido es de 0 a 255, lo que utiliza 8 bits. Si el tipo de datos que se está transfiriendo es texto simple (ASCII estándar), entonces es suficiente con utilizar 7 bits por paquete para la comunicación.

c. **Bits de parada:** Usado para indicar el fin de la comunicación de un solo paquete. Los valores típicos son 1, 1.5 o 2 bits. Debido a la manera como se transfiere la información a través de las líneas de comunicación y que cada dispositivo tiene su propio reloj, es posible que los dos dispositivos no estén sincronizados. Por lo tanto, los bits de parada no sólo indican el fin de la transmisión sino además dan un margen de tolerancia para esa diferencia de los relojes. Mientras más bits de parada se usen, mayor será la tolerancia a la sincronía de los relojes, sin embargo la transmisión será más lenta.

d. **Paridad:** Es una forma sencilla de verificar si hay errores en la transmisión serial. Existen cuatro tipos de paridad: par, impar, marcada y espaciada. La opción de no usar paridad alguna también está disponible.

Para paridad par e impar, el puerto serial fijará el bit de paridad (el último bit después de los bits de datos) a un valor para asegurarse que la transmisión tenga un número par o impar de bits en estado alto lógico. Por ejemplo, si la información a transmitir es 011 y la paridad es par, el bit de paridad sería 0 para mantener el número de bits en estado alto lógico como par. Si la paridad seleccionada fuera impar, entonces el bit de paridad sería 1, para tener 3 bits en estado alto lógico.

La paridad marcada y espaciada en realidad no verifican el estado de los bits de datos; simplemente fija el bit de paridad en estado lógico alto para la marcada, y en estado lógico bajo para la espaciada. Esto permite al dispositivo receptor conocer de antemano el estado de un bit, lo que serviría para determinar si hay ruido que esté afectando de manera negativa la transmisión de los datos, o si los relojes de los dispositivos no están sincronizados.

2.7.2 Funcionamiento de Protocolo Serial

Para utilizar el puerto serie se necesita, además de hardware, un software que realice el control del dispositivo. Este software es ofrecido por el BIOS y concretamente por la interrupción 14H la cual contiene cuatro funciones que permiten este control

- Función 01H: Enviar carácter
- Función 02H: Leer carácter
- Función 03H: Obtener estado
- Función 00H: Inicialización

El protocolo es el modo de codificar los datos que van a enviarse, para obtener un entendimiento entre los dos puntos participantes de la transmisión. Para este protocolo solo importan dos estados

0 = Bajo

1 = Alto

Si no se está transmitiendo ningún carácter, la línea esta en un estado alto, mientras que si se transmiten algún carácter estará en estado bajo.

2.7.3 Circuito Controlador de Puerto Serial MAX232

Es un circuito integrado que convierte los niveles de las líneas de puerto serie RS-232 a niveles TTL y viceversa. Lo interesante es que sólo necesita una alimentación de 5V, ya que genera internamente algunas tensiones que son necesarias para el estándar RS-232.

El MAX232 soluciona la conexión necesaria para lograr comunicación entre el puerto serie de una PC o cualquier otro circuito con funcionamiento en base a señales de nivel TTL/CMOS. Estos conversores son suficientes para manejar las cuatro señales más utilizadas del puerto serie, que son TX (señal de transmisión de datos), RX (es la de recepción), RTS y CTS (establecen el protocolo para el envió y recepción de los datos).

El circuito integrado necesita para funcionar sólo de cuatro capacitores electrolíticos y de una fuente de alimentación de 5V, internamente el circuito tiene dos fuentes conmutadas, la primera de ellas en conjunto con los capacitores electrolíticos y “adaptan” el nivel de voltaje tomado de la alimentación de +5V a +10V, la segunda fuente conmutada y los capacitores electrolíticos invierten los niveles de voltaje para que se pueden obtener -10V, estos niveles de voltaje son utilizados para realizar la adaptación de los voltajes RS-232.

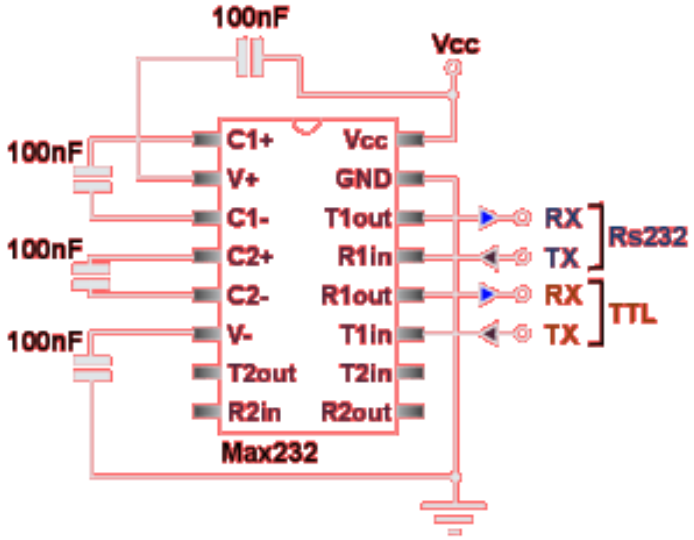


Figura 2.8 – Circuito Max232

Cabe mencionar que adema del MAX232, existe una gran variedad de circuitos integrados que cumplen con la norma RS-232 como son: DS14C232, MAX220, LT1180A, MAX220. A continuación se muestra una tabla donde se detallan algunas de sus características principales así como el costo de adquisición de cada componente:

CI que cumplen con la norma RS-232	**MAX232**	**DS14C232**	**MAX220**	**LT1180A**	**MAX233**	**MAX3232C**
Distribuidor	Texas Instruments Maxim Dallas Semiconductor	National Semiconductor	Texas Instruments -MAXIM Dallas Semiconductor	Linear Tecnology	MAXIM Dallas Semiconductor	Texas Instruments
Suministro de Voltaje	+5 (V)	4.5 - 5.5 (V)	+5 (V)	+5 (V)	+5 (V)	+3.3– 5(V)
No. De Conductores/ Receptores	2/2	2/2	2/2	2/2	2/2	2/2
Empaquetado (No. de pines)	16	16	16	16/18	30	16
Precio por unidad (Pesos)	11-20	28	11-20	89.50	38	30

Tabla 2.2 Tabla comparativa de CI que cumplen con la norma RS-232 ***[26]***

2.8 Protocolo Ethernet

Es un estándar de transmisión de datos para redes de área local y es probablemente el estándar más popular. A fines de 1996 más del 80% de las redes instaladas en el mundo eran Ethernet.
Fue desarrollado inicialmente en 1973 por el Dr. Robert M. Metcalfe en el PARC (Palo Alto Research Center) de la compañía Xerox, como un sistema de red denominado Ethernet Experimental. El objetivo era conseguir un medio de comunicación entre computadoras.

Estos primeros trabajos del PARC contribuyeron substancialmente a la definición de la norma IEEE 802.3, que define el método de acceso CSMA/CD. Esté se basa en que cuando un equipo DTE ("Data Terminal Equipment") conectado a una LAN desea transmitir, se mantiene a la escucha hasta que ningún equipo está transmitiendo, una vez que la red está en silencio, el equipo envía el primer paquete de información.

Los estándares Ethernet no necesitan especificar todos los aspectos y funciones necesarios en un Sistema Operativo de Red.

La especificación Ethernet se refiere solamente a las dos primeras capas del modelo OSI:

- La capa física: el cableado y las interfaces físicas.
- La de enlace: que proporciona direccionamiento local, detección de errores, y controla el acceso a la capa física. **[21]**

IEEE:

Corresponde a las siglas de *The Institute of Electrical and Electronics Engineers*, el Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial fundada en 1884 y dedicada a la estandarización.

NORMA IEEE 802

IEEE 802S es un comité y grupo de estudio de estándares perteneciente al Instituto de Ingenieros Eléctricos y Electrónicos que actúa sobre Redes de Ordenadores, concretamente y según su propia definición sobre redes de área local (LAN) y redes de área metropolitana (MAN)

La IEEE 802 es el nombre de un comité de estandarización del IEEE y por su extensión se denomina así los estándares por él producidos.

La primera versión fue un intento de estandarizar Ethernet aunque hubo un campo de la cabecera que se definió de forma diferente. Posteriormente ha habido aplicaciones sucesivas al estándar que cubrieron las aplicaciones de velocidad, redes virtuales, hubs, conmutadores y distintos tipos de medios, tanto de fibra óptica como de cables de cobre (tanto de par trenzado como coaxial).

Los estándares de este grupo no reflejan necesariamente o que usa en la práctica aunque a diferencia de otros grupos este suele estar cerca de la realidad.

2.8.1 Controladores Ethernet

Es un dispositivo autónomo, con una interfaz periférica serial estándar industrial (SPI), está diseñado para ser utilizado como una interfaz de red Ethernet para cualquier controlador equipado

A continuación se presentan tres tipos de controladores Ethernet y sus respectivas características. **[27]**

Controlador ENC28J60

- Controlador compatible con el estándar IEEE 802.3
- Soporta la capa MAC (Control Acceso al Medio) y Física con el estándar 10Baste-T.
- Soporta los modos de comunicación Full y Hall Dúplex.
- Reenvió automático programable de paquetes erróneos.
- Interfaz SPI (interfaz periférica serial estándar industrial) con velocidades de hasta 10Mbits.

Controlador RTL8111C

- Compatible con IEEE 802.3
- Soporta la capa MAC (Control Acceso al Medio)
- Un controlador de bus PCI (Interconexión de Componentes Periféricos)
- Tecnología DSP (Procesamiento Digital de Señales)
- Corrección de polaridad y corrección de errores.
- Soporta Full-Dúplex.
- Soporta protocolos IPv4, IPv6, UDP (Protocolo de datagrama de usuario)

Controlador WIZ W5100

- Con interfaz SPI.
- Capa MAC y PHY (Capa Física)
- Soporta los protocolos TCP (Protocolo de Control de Transmisión), UDP (Protocolo de Datagramas de usuario), ARP (Protocolo de Resolución de Direcciones), IPv4.
- Soporta Full y Half Dúplex.

2.9 Protocolo de Red

Se conoce como protocolo de comunicaciones a un conjunto de reglas que especifican el intercambio de datos durante la comunicación entre ordenadores.

2.9.1 Modelo OSI

El modelo de Interconexión de Sistemas Abiertos, está basado en una propuesta desarrollada por la Organización Internacional de Normas (ISO), como primer paso hacia la normalización internacional de varios protocolos. **[28]**

Todas las interfaces de comunicación se basan en el modelo OSI y el protocolo que rige nuestro trabajo nos define una serie de reglas y primitivas que permiten a una red intercambiar información, sean estas locales o de área extensa.

Este modelo se puede definir en siete niveles o capas, que son: **[29]**

1. Capa Física

Dicha capa está encargada de la interfaz física entre los dispositivos, así como de definir las reglas que rigen en la transmisión de los bits, es decir se encarga de transformar un paquete de información binaria ("Frame") en una sucesión de impulsos adecuados al medio físico utilizado en la

transmisión. Estos impulsos pueden ser eléctricos (transmisión por cable); electromagnéticos (transmisión Wireless) o luminosos (transmisión óptica). Cuando actúa en modo recepción el trabajo es inverso; se encarga de transformar estos impulsos en paquetes de datos binarios que serán entregados a la capa de enlace

Figura 2.9 Capas modelo OSI

2. Capa de Enlace de datos

Puede decirse que esta capa traslada los mensajes hacia/desde la capa física a la capa de red. Especifica cómo se organizan los datos cuando se transmiten en un medio particular.

Además del direccionamiento local, se ocupa de la detección y control de errores ocurridos en la capa física, del control del acceso a dicha capa y de la integridad de los datos y fiabilidad de la transmisión. Para esto agrupa la información a transmitir en bloques ("Frames"). Los datagramas recibidos son comprobados por el receptor. Si algún datagrama se ha corrompido se envía un mensaje de control al remitente solicitando su reenvío.

3. Capa de Red

Esta capa se ocupa de la transmisión de los datagramas (paquetes) y de encaminar cada uno en la dirección adecuada ("Routing"), tarea esta que puede ser complicada en redes grandes como Internet, pero no se ocupa para nada de los errores o pérdidas de paquetes. Como consecuencia esta capa puede considerarse subdividida en dos:

- Transporte. Encargada de encapsular los datos a transmitir (de usuario). Utiliza los paquetes de datos. En esta categoría se encuentra el protocolo IP ("Internet Protocol").
- Conmutación. Esta parte es la encargada de intercambiar información de conectividad específica de la red.

4. Capa de Transporte

Esta capa se ocupa de garantizar la fiabilidad del servicio, describe la calidad y naturaleza del envío de datos, esta capa define cuando y como debe utilizarse la retransmisión para asegurar su llegada. Para ello divide el mensaje recibido de la capa de sesión en trozos (datagramas), los numera correlativamente y los entrega a la capa de red para su envío. Durante la recepción, si la capa de Red utiliza el protocolo IP, la capa de Transporte es responsable de reordenar los paquetes recibidos fuera de secuencia.

5. Capa de Sesión

Es una extensión de la capa de transporte que ofrece control de diálogo y sincronización, aunque en realidad son pocas las aplicaciones que hacen uso de ella. Por ejemplo, las comunicaciones de Internet no la utilizan.

6. Capa de Presentación

Esta capa se ocupa de los aspectos semánticos de la comunicación (describe la sintaxis de los datos a transmitir) estableciendo los arreglos necesarios para que puedan comunicar máquinas que utilicen diversa representación interna para los datos, describe como pueden transferirse números de coma flotante entre equipos que utilizan distintos formatos matemáticos.

En teoría esta capa "presenta" los datos a la capa de aplicación atrapando los datos recibidos y transformándolos en formatos como texto imágenes y sonido. En realidad esta capa puede estar ausente, ya que son pocas las aplicaciones que hacen uso de ella.

7. Capa de Aplicación

Esta capa describe como hacen su trabajo los programas de aplicación (navegadores, clientes de correo, terminales remotos, transferencia de ficheros etc.). Por ejemplo, esta capa implementa la operación con ficheros del sistema. Por un lado interactúan con la capa de presentación; por otro representan la interfaz con el usuario, entregándole la información y recibiendo los comandos que dirigen la comunicación.

Para nuestro caso en particular se tiene una Pila TCP/IP de Microchip (Una pila TCP/IP es un conjunto de rutinas programadas para implementar los protocolos de Internet), junto con una librería del Puerto Serial.

La pila de Microchip es de código abierto, por lo que es posible modificarlo y recompilarlo cuantas veces sea necesario, y adaptarlo a nuestras necesidades. La pila es modular y flexible, lo que permite activar y desactivar módulos, como también usar páginas web dinámicas que permiten controlar todos los recursos del PIC remotamente, usando HTTP por ejemplo.

2.9.2 TCP/IP

La arquitectura de protocolos TCP/IP es resultado de la investigación y desarrollo llevados a cabo en la red experimental de conmutación de paquetes ARPANET y se denomina globalmente como la familia de protocolos TCP/IP. Esta familia de protocolos se ha especificado como estándares de Internet. **[28]**

El Modelo TCP/IP estructura el problema de la comunicación en cinco capas relativamente independientes entre sí: **[28]**

- Capa física: Define la interfaz física entre el dispositivo de transmisión de datos y el medio de transmisión o red.

- Capa de acceso a la red: Es responsable del intercambio de datos entre el sistema final (servidor, estación de trabajo, etc.) y la red a la cual está conectado, es decir está relacionada con el acceso y encaminamiento de los datos.

- Capa de Internet: El protocolo de Internet (IP, *Internet Protocol*) se utiliza e esta capa para ofrecer el servicio de encaminamiento a través de varias redes.

- Capa extremo-a-extremo o de trasporte: El protocolo para el control de la transmisión, TCP (*Transmission Control Protocol*), es el más utilizado para proporcionar la funcionalidad de agrupar todos los mecanismo en una capa común compartida por todas las aplicaciones.

- Capa de aplicación: Contiene toda la lógica necesaria para posibilitar las distintas aplicaciones de usuario.

2.9.3 Comparación entre OSI y TCP/IP

El modelo OSI y el TCP/IP tienen muchas cosas en común. Ambos se basan en la idea de una pila de protocolos independientes. Además, la funcionalidad de las capas es bastante similar. Por

ejemplo en ambos modelos, las capas hasta la de transporte deben proporcionar un servicio a procesos que deseen comunicarse.

Aún así, también poseen muchas diferencias. El modelo OSI tiene tres conceptos básicos: servicios, interfaces y protocolos. Probablemente, la principal contribución del modelo OSI es hacer explícita la distinción entre sus conceptos. Cada capa realiza unos servicios para la capa superior.

En su origen, el modelo TCP/IP no hizo esta distinción, aunque con el tiempo se ha adecuado a los propuestos por el modelo OSI.

El modelo OSI se planteó antes de definir los protocolos de cada capa por ello el modelo no se desvió a favor de ningún protocolo. Con TCP/IP sucedió lo inverso: primero se definieron los protocolos y modelado resulto ser una descripción de los mismos.

Eficiencia y viabilidad. Las normas de OSI tienden a ser prescriptivas (por ejemplo, la capa "N" debe atravesar todas las capas "por debajo" de ella), mientras que los protocolos TCP/IP tienden a ser descriptivos, y dejan un máximo de libertad a los implementadores. **[30]**

2.10 Modelos de Red.

2.10.1 Modelo Cliente/Servidor

En este modelo de trabajo, cuando una terminal (cliente) pide datos al servidor, la información es procesada por un programa que se ejecuta en el servidor de archivos y monitorea todo el tiempo las peticiones de archivos de datos (servidor de datos). El servidor de datos localiza la información pedida y solamente la información de respuesta es la que viaja por la red.

El servidor de datos es el encargado de la apertura de la base de datos, la cual se abre una vez sin importar el número de clientes que estén accesando a los archivos. Si se presenta alguna falla, el servidor es el encargado de controlar que no se cierre el acceso a los archivos utilizados por los demás clientes. **[31]**

Las características más importantes que se distinguen cliente/servidor son: **[32]**

- Orientado a servicios. El servidor los ofrece y el cliente los consume.
- Compartición de recursos. Servicios ofrecidos a muchos clientes. Un servidor puede atender muchos clientes que solicitan esos servicios.
- Transparencia de ubicación. El servidor es un proceso que puede residir en el mismo aparato que el cliente o en un aparato distinto a lo largo de una red. Un programa puede ser un servidor en un momento y convertirse en un cliente posteriormente.

- Mezcla e igualdad. Tal vez de las más importantes ventajas de este paradigma. Una aplicación cliente/servidor, idealmente es independiente del hardware y de sistemas operativos; mezclando e igualando estas plataformas.
- Interacción a través de mensajes, para envío y respuesta de servicios.
- Servicios encapsulados, exponiendo los servicios a través de interfaces, lo que facilita la sustitución de servidores son afectar los clientes; permitiendo a la vez una fácil escalabilidad.

2.10.2 Modelo Vista Controlador

Este modelo es un patrón de arquitectura que separa los datos de aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Este patrón es frecuentemente aplicado en páginas web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página.

El modelo es todo acceso a datos, cada acceso a datos se pone en función individual debido a que sí se cambia de SGBD, el cambio sólo afecta a estas funciones y no al resto de la aplicación.

La vista, es una aplicación web, es el HTML y lo necesario para convertir datos en HTML. En otras palabras, muestra la información del modelo al usuario.

El controlador es lo que une la vista y el modelo. Por ejemplo, las funciones que toman los valores de un formulario, consultan la base de datos, a través del modelo, y producen valores, que la vista tomará y convertirá en HTML. **[33]**

En resumen, el controlador gestiona las entradas del usuario, recibe los eventos de entrada y contiene reglas de gestión de eventos. Estas acciones pueden suponer peticiones al modelo o a las vistas. De este modo, el código que realiza una acción, está perfectamente separado del código dedicado a crear HTML, tal como se muestra en la figura 2.10.

2.10.3 Programación por capas o Modelado a capas

Al construir un software se llevan a cabo varias técnicas de desarrollo para que el mismo se haga en forma ordenada, lo cual beneficia en cuanto a reducción de costos por tiempo, debido a que se podrá avanzar de manera más segura en el desarrollo. (ver figura 2.11)

La programación por capas es una técnica de ingeniería de software que se organiza principalmente en 3 capas: la capa de presentación o frontera, la capa lógica de negocio o control y la capa de datos. **[34]**

1. Capa de presentación.

   El objetivo de esta capa es facilitar al usuario la interacción con la aplicación. La interfaz debe ser amigable y fácil de utilizar.

   Las interfaces deben de ser consistentes, específica y satisfacer los requerimientos del usuario.

Esta capa contiene los objetos encargados de comunicar al usuario con el sistema mediante el intercambio de información.

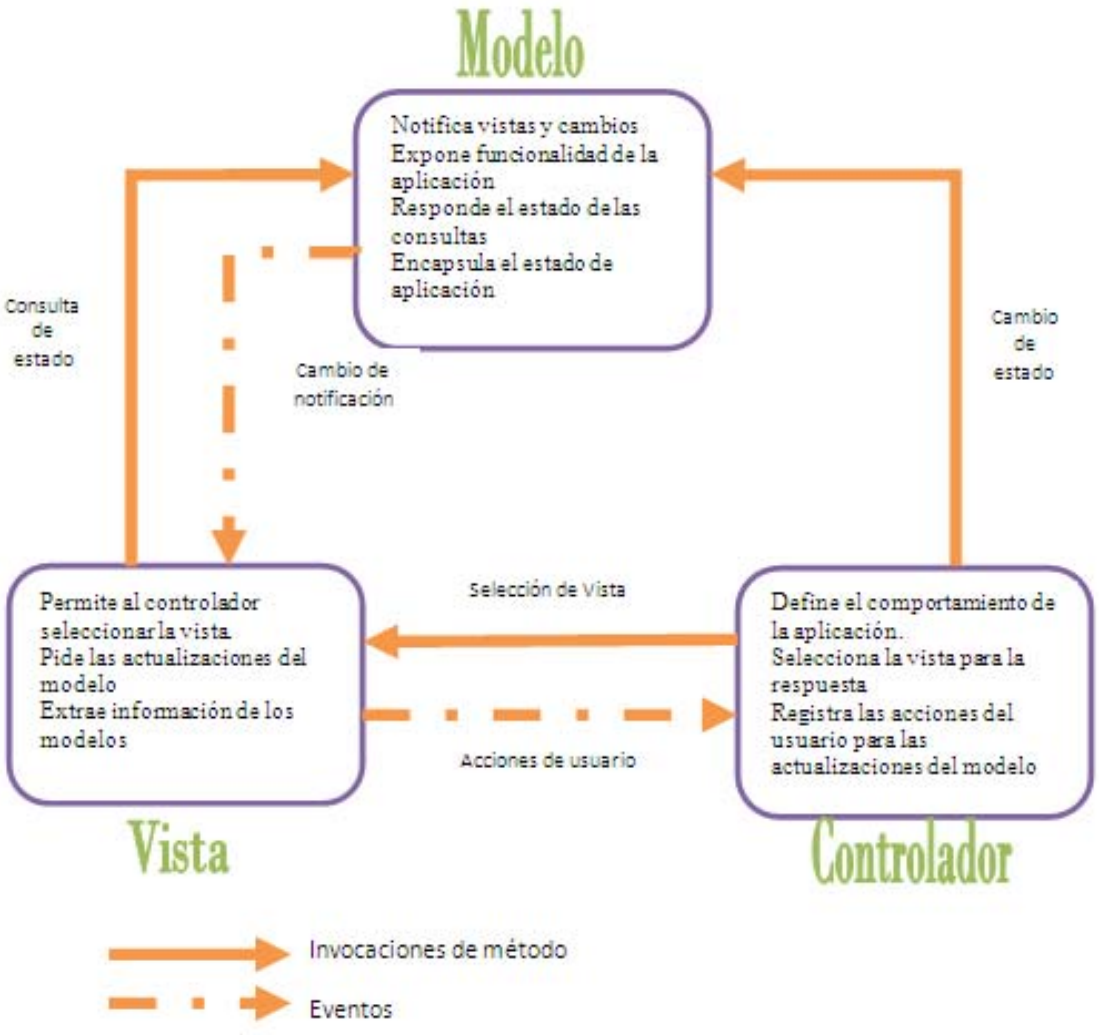


Figura 2.10 Modelo Vista Controlador **[33]**

2. Capa de negocio.

   Comunica a todas las demás capas para poder llevar a cabo las tareas. Es llamada capa de las reglas de negocio porque en esta se definen todas las reglas que se deben cumplir para una correcta ejecución del programa.

   En esta capa se encuentran las estructuras de datos y objetos encargados para la manipulación de los datos existentes.

3. Capa de datos.

   Es donde residen los datos y es la capa encargada de acceder a los mismos. El manejo de los datos debe realizarse de forma tal que haya consistencia en los mismos, de tal forma los datos que se ingresan y/o extraen deben ser consistentes y precisos.

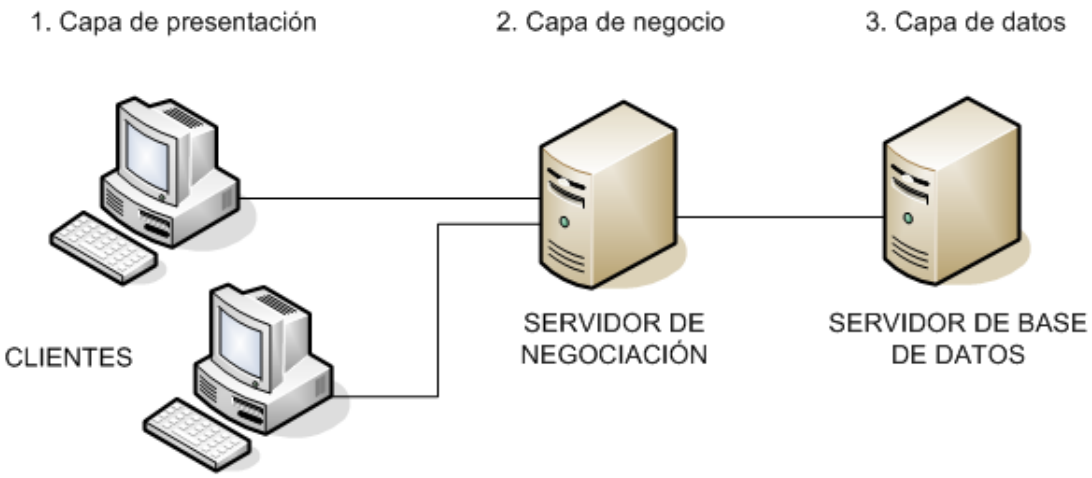


Figura 2.11 Modelado a tres capas

Un modelado a capas de dos niveles es representado por la capa de presentación y la capa de datos, quedando la interfaz gráfica y la lógica de negocios dentro de la capa de presentación y los datos residen dentro de la capa de datos.

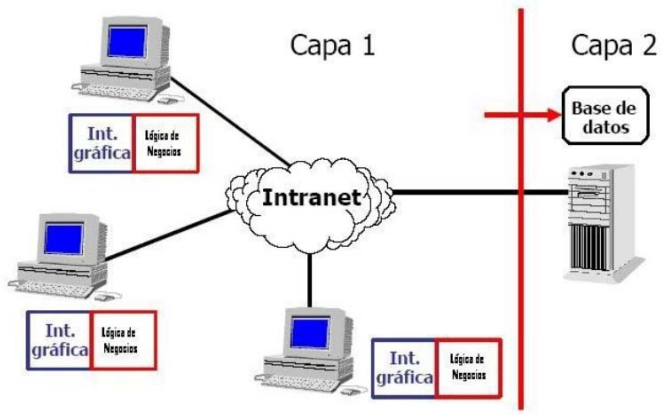


Figura 2.12 Modelado a dos capas

2.11 Servidor de Aplicaciones

Un servidor de aplicaciones es un software que proporciona aplicaciones a los equipos o dispositivos clientes, generalmente a través de Internet siguiendo el protocolo HTTP. Los servidores de aplicación se distinguen de los servidores web por el uso del contenido dinámico y por la frecuente integración con base de datos. **[35]**

Características

Generalmente un servidor de aplicación incluye un middleware, que es un software de que trabaja como intermediario para la seguridad y mantenimiento, además de proveer acceso a los datos.

Principios fundamentales: **[36]**

- Alta disponibilidad. Funcionamiento las 24 horas de día los 365 días del año.
- Escalabilidad. Capacidad de hacer crecer un sistema.
- Mantenimiento. Versatilidad a la hora de depurar fallos y mantener un sistema.

Ventajas: **[37]**

- Integridad de datos y códigos. Al encontrarse centralizada, una o más máquinas las actualizaciones siempre se realizan para todos los usuarios.
- Configuración centralizada. Los cambios de configuración de la aplicación así como mover la base de datos se puede realizar centralmente.
- Seguridad.
- Performance. Limitando el tráfico de la red en la capa de presentación, es percibido como un modelo cliente/servidor que mejora el performance de grandes aplicaciones.

2.11.1 IIS (Internet Information Services)

Internet Information Services (IIS) es un potente servidor Web que ofrece una infraestructura de gran fiabilidad, capacidad de manejo y escalabilidad para aplicaciones Web.

La versión 6.0 de IIS soporta sistemas dinámicos capaces de auto-administrarse, con monitorización automática, aislamiento de procesos y capacidades de gestión mejoradas.

En la actualidad existe IIS 7.0. La configuración del servidor web es almacenada en archivos XML y el manejo de una cuenta lógica que posee menos privilegios que la cuenta de administrador; sin embargo, la hace más segura son una de las nuevas características de esta versión.

Las arquitecturas de las versiones anteriores a la 6 se consideran como servidores monolíticos, de la versión 6 en adelante se desarrolla un tipo de arquitectura modular para los servicios. Con esta arquitectura surgen funcionalidades más específicas, parecidas a las que realiza el servidor Apache. La ventaja de utilizar este tipo de arquitectura es que sólo son habilitadas las características requeridas y las funcionalidades pueden ser ampliadas creando módulo mejorando el rendimiento y la seguridad. **[38]**

La versión 7 de IIS incluye:

- Módulos manejadores de peticiones HTTP.
- Módulos de seguridad.
- Módulos de contenidos.
- Módulos para conversión/ compresión de información.
- Módulos de caché.

- Módulos de registro y diagnóstico de peticiones.
- API para programar nuevos módulos y funcionalidades para el servidor, utilizando lenguaje .NET.

2.11.2 Apache

Servidor web de código abierto para plataformas UNIX(BSD, GNU/Linux), Windows, Macintosh y otras que implementa el protocolo HTTP. Su desarrollo comenzó en febrero de 1995, por Rob McCool. La primera versión apareció en enero de 1996, el Apache 1.0. **[39]**

El servidor Apache se desarrolla dentro del proyecto HTTP Server (httpd) de la Apache Software Foundation. El nombre «Apache» es un acrónimo de «a patchy server» -un servidor parchado-, es decir un servidor construido con código preexistente, piezas y parches de código.

Hacia el 2000, el servidor Web Apache era el más extendido en el mundo y las principales metas de su diseño son: velocidad, simplicidad, multiplataforma y facilidad de desarrollo distribuido.

CARÁCTERÍSTICAS	**SERVIDORES WEB**	
	APACHE	**IIS**
Software libre	SI	NO
Capacidad de ser empotrado	NO	NO
Conexiones permanentes	SI	SI
Módulos/Soporte plug-ins	SI	SI
Soporte Virtual Servers	SI	SI
Escala a servidores SMP	SI	SI
Autenticación	SI	SI
Soporte Encoders	SI	SI
CGI's	SI	SI
Páginas de error personalizadas	SI	SI
Conexiones seguras https	SI	SI
Caché friendly	SI	SI
Líneas de código	185 000	¿?

Tabla 2.3 Comparativa entre servidores web

2.12 Modelo de Base de Datos

Un modelo de datos es la colección de herramientas para describir los datos, sus relaciones, su semántica y las restricciones de consistencia. Estos modelos nos describen el diseño de las bases en el nivel físico, lógico y de vistas.

Niveles de las bases de datos: **[40]**

El nivel físico es el nivel más bajo de abstracción, en él se describe cómo se almacenan realmente los datos.

El nivel lógico *describe qué datos se almacenan en la base de datos y que relaciones existen entre esos datos*

Nivel de vistas *es el nivel más alto de abstracción describe sólo parte de la base de datos completa. Los usuarios necesitan acceder sólo a una parte de la base de datos. El sistema puede proporcionar muchas vistas para la base de datos.*

El modelo de base de datos puede clasificarse en cuatro categorías diferentes: **[41]**

- Modelo relacional
- Modelo entidad-relación
- Modelo de datos orientado a objetos
- Modelo de datos semiestructurados.

2.12.1 Modelo Relacional

El modelo relacional fue propuesto por E.F. Codd en 1970 por los laboratorios de IBM en California.

Las bases de datos relacionales usan un conjunto de tablas para representar tanto los datos como las relaciones entre ellos. Utilizan un lenguaje de definición de datos (LDD), para especificar el esquema de la base de datos, y un lenguaje de manipulación de datos (LMD), para consultas y modificaciones de la base de datos.

Este modelo permite representar la información del mundo real de una manera intuitiva, introduciendo conceptos cotidianos y fáciles de entender. Asimismo, mantiene información sobre las propias características de la base de datos (metadatos), que facilitan las modificaciones, disminuyendo los problemas ocasionados en las aplicaciones ya desarrolladas.

Una base de datos relacional consiste en un conjunto de tablas cada una de ellas con un nombre exclusivo. Cada fila representa una relación entre un conjunto de valores, es decir, cada tabla es un conjunto de entidades, y cada fila es una entidad. **[42]**

2.12.2 Modelo Entidad-Relación (E-R)

El modelo de datos entidad-relación está basado en un conjunto de objetos básicos, denominados entidades, y de las relaciones entre esos objetos. Una entidad es una “cosa” u “objeto” del mundo real que es distinguible de otros objetos. Las entidades se describen en la bases de datos mediante un conjunto de atributos, un atributo es una característica que posee una entidad, y que agrupadas permiten distinguirla entre otras.

El conjunto de todas las entidades del mismo tipo, y el conjunto de todas las relaciones del mismo tipo se denominan, respectivamente, conjunto de entidades y conjunto de relaciones.

Una relación es una asociación entre varias entidades. Dentro de esta relación existe el concepto de grado, que es el número de conjuntos de entidades que intervienen en una interrelación. **[43]**

Dependiendo del número de entidades del primer conjunto de entidades y del segundo existen tres tipos de interrelaciones binarias, las cuales son: **[42]**

- Uno a uno (1:1)

  Si tenemos dos conjuntos de entidades A y B respectivamente. Cada entidad A se asocia con una entidad de B y cada entidad en B se asocia con una entidad de A.

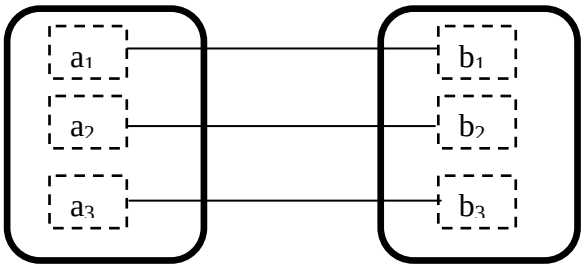


Figura 2.13 Modelo E-R Uno a uno

- Uno a muchos (1:N), muchos a uno (N:1)

  Cada entidad de A se asocia con cualquier número de entidades de B; sin embargo, cada entidad de B se puede asociar con una entidad de A

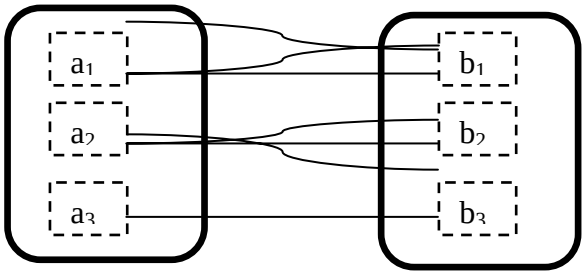


Figura 2.14 Modelo E-R Uno a muchos

- Muchos a muchos (N:M)

  Cada entidad de A se asocia con cualquier número (cero o más) de entidades de B y cada entidad B se asocia con cualquier número de entidades de A.

En principio, cada entidad se puede distinguir de otra por sus atributos. Aunque un subconjunto pueda ser iguales en entidades distintas, el conjunto completo de todos los atributos no se puede repetir nunca.

Es necesario tener una forma para distinguir las entidades pertenecientes a un conjunto de entidades dado. Cada entidad es distinta, la diferencia entre ellas se expresa en términos de sus atributos.

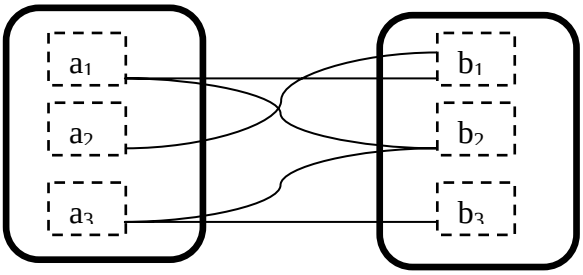


Figura 2.15 Modelo E-R Uno a muchos

Las claves permiten identificar un conjunto de atributos para distinguir las entidades entre sí.

Una superclave es un conjunto de uno o más atributos que permiten identificar una entidad del conjunto de entidades.

Se le llama clave primaria o llave primaria al atributo considerado clave para la identificación de los demás atributos que describen a la entidad.

Puede existir más de un atributo que pueda identificarse como llave primaria, en este caso se selecciona a la que se considere más importante, las demás son denominadas llaves secundarias.
Una llave o clave primaria es identificada gráficamente en el modelo E-R con una línea debajo del nombre del atributo.

La estructura lógica general de la base de datos se puede expresar gráficamente mediante un diagrama E-R, que está constituido por los siguientes componentes: **[42]**

- Rectángulos, que representan conjuntos de entidades.
- Elipses, que representan atributos
- Rombos, que representan conjuntos de relaciones entre miembros de varios conjuntos de entidades.
- Líneas, que unen los atributos con los conjuntos de entidades entre sí, y también los conjuntos de entidades con las relaciones.
- Elipses dobles, que representan atributos multivalorados.
- Elipses discontinuas, que denotan atributos derivados
- Líneas dobles, que indica participación total de una entidad en un conjunto de relaciones
- Rectángulos dobles, que representan conjuntos de entidades débiles

Cada componente se etiqueta con la entidad o relación que representa.

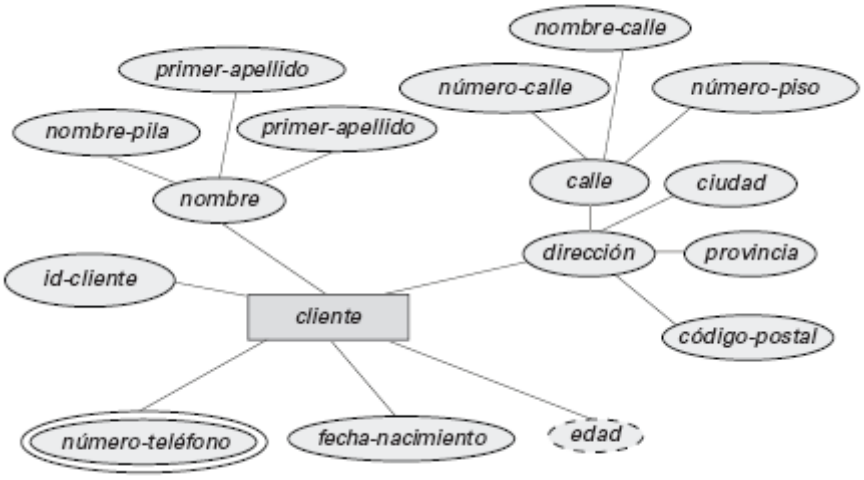


Figura 2.16 Ejemplo de diagrama E-R

Normalización

El objetivo de la normalización es generar un conjunto de esquemas de relaciones que permita almacenar información y garantizar la calidad de los elementos además de la seguridad.

Grados de normalización

Existen básicamente tres niveles de normalización: **[41]**

- Primera Forma Normal (1FN). Cada columna debe ser atómica. **[43][44]**
  Es decir, cada atributo debe contener un único valor y nombre. Para aplicar esta forma normal basta con dividir cada columna no atómica en tantas columnas atómicas como sean necesarias.
- Segunda Forma Normal (2FN). Debe cumplir primero con la 1FN **[45]**
  Para poder explicar esta forma de normalización debemos definir dependencia funcional.
  La dependencia funcional consiste en elegir que atributos dependen de otro(s) atributo(s).
  Una relación está en 2FN si y solo si está en 1FN y los atributos no llaves dependen por completo de la clave.
- Tercera Forma Normal (3FN). Debe cumplir primero con la 2FN
  Consiste en eliminar la dependencia transitiva entre los atributos. Una dependencia transitiva es cuando existe más de una forma de llegar a referencias a un atributo de una relación.

2.12.3 Modelo Orientado a Objetos

El modelo de datos relacional orientado a objetos (objeto-relación) puede considerarse una extensión del modelo E-R ofreciendo un sistema más rico que incluye tipos de datos complejos y orientación a objetos (encapsulación, métodos e identidad de los objetos).

Este tipo de modelo se desarrolló debido a la demanda que existe para abordar tipos de datos más complejos. Este tipo de modelo permite tener una extensión de los lenguajes de programación y los sistemas relacionales orientados a objetos. Gracias a esta característica este modelo ofrece un medio de migración cómodo para los usuarios de las bases de datos relacionales que deseen usar las características del modelo objeto-relación.

Como se mencionó anteriormente, este modelo se basa en el paradigma de los lenguajes de programación orientados a objetos: herencia, identidad de los objetos y la encapsulación, con métodos para ofrecer una interfaz para los objetos.

El término de sistemas de bases de datos orientadas a objetos se usa para hacer referencia a los sistemas de bases de datos que soportan sistemas de tipos orientados a objetos y permiten el acceso directo a los datos desde los lenguajes de programación orientados a objetos usando el sistema de tipos nativo del lenguaje. **[41]**

2.12.4 Modelo de datos semiestructurados

Permiten la especificación de los datos en los que cada elemento de datos del mismo tipo puede tener conjuntos de atributos diferentes. La diferencia entre los otros modelos es que todos los elementos de datos de un tipo en específico deben tener el mismo conjunto de atributos.

El lenguaje XML (Extensible Markup Language) se diseñó inicialmente como un modo de añadir información de marcas a los documentos de texto, pero se ha vuelto importante debido a sus aplicaciones en el intercambio de datos. XML, ofrece un modo de representar los datos que tienen una estructura anidada y, además, permite una gran flexibilidad en la estructuración de los datos.

Comparando el almacenamiento de los datos de XML con la de una base de datos relacional, XML puede parecer poco eficiente debido a que los nombre de las etiquetas se repiten por todo el documento; sin embargo, tiene varias ventajas cuando se usa para intercambio de datos y para almacenar información estructurada en archivos, estas son: **[41]**

- Autodocumentado. La presencia de etiquetas hace un esquema fácil de comprender el significado del texto
- Formato no rígido. Si se agrega información adicional, el recipiente de datos WML ignora la etiqueta. La etiqueta se requiere en los elementos que se ordenan por peso o volumen, y se pueden omitir en elementos que se ordenan por número. Esta capacidad de reconocer e ignorar etiquetas permite al formado de los datos evolucionar con el tiempo sin invalidar las aplicaciones

existentes. La posibilidad de que una etiqueta aparezca varias veces hace sencillo representar atributos multivalorados.

- Permite estructuras anidadas. Es decir cada elemento tiene a su vez otro anidado, por ejemplo; en un pedido de compra, un comprador tiene un nombre y dirección anidados.
- Variedad de herramientas de ayuda. Puesto que el formato XML es aceptado ampliamente existen herramientas disponibles para ayudar a su procesamiento, incluyendo APIs para lenguajes de programación para crear y leer datos XML, software de exploración y herramientas de bases de datos.

2.13 Manejadores de Bases de Datos

2.13.1 Informix

Informix fue desarrollado por Roger Sippl a finales de 1970, en 1990 fue el segundo sistema de bases de datos más popular, después de Oracle; sin embargo, para el año 2000 debido a fallas de gestión decayó su popularidad. En el año 2001 IBM compró Informix. Su última versión surgió a principios de 2005, llamada IDS (Informix Dynamic Server) 11.5. **[46]**

Informix ofrece alta disponibilidad implementando hardware de bajo costo. Cada servidor puede ser utilizado para distribuir la carga de trabajo, proporcionando mayor potencia de procesamiento para obtener el máximo provecho del hardware.

IDS 11.5 es la base de datos ideal para la gestión de múltiples bases de datos con los requisitos mínimos de administración de bases de datos.

Las características que incluye la nueva versión son:**[47]**

- Compresión de datos.
- Depósito de datos.
- Tecnología embebida.
- Optimización de almacenamiento.

2.13.2 Oracle

Oracle es un sistema de administración de base de datos (o RDBMS Relational Data Base Management System por las siglas en inglés), fabricado por Oracle corporation. Sus orígenes se remontan a finales de los 70 bajo el nombre de Relational Software a partir de un estudio realizado por George Koch. Su última versión es la 11g, liberada en el año 2007.

Oracle es una herramienta cliente/servidor para la gestión de Bases de Datos el cual es vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales. En el desarrollo de páginas web pasa lo mismo: como es un sistema muy caro no está tan extendido como otras bases de datos.

Cada actualización desde Oracle 8 hasta 11g ha incrementado el rendimiento, estabilidad y disponibilidad del sistema.

Para desarrollar en Oracle utilizamos PL/SQL un lenguaje de 5ª generación, bastante potente para tratar y gestionar la base de datos, también por norma general se suele utilizar SQL.

Oracle es sin duda una de las mejores bases de datos que tenemos en el mercado, es un sistema gestor de base de datos robusto, tiene muchas características que nos garantizan la seguridad e integridad de los datos; que las transacciones se ejecuten de forma correcta, sin causar inconsistencias; ayuda a administrar y almacenar grandes volúmenes de datos; estabilidad, escalabilidad y es multiplataforma. [49]

2.13.3 PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD (Berkeley Software Distribution) y con su código fuente disponible libremente. Se comenzó a desarrollar en 1996 tal cual lo conocemos ahora, aunque sus principios sean en los años 70. Su última versión disponible es la 8.4.0.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

Las características de este gestor lo hacen uno de los más potentes, estables y robustos del mercado. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. **[49]**

Límite	Valor
Máximo tamaño base de datos	Ilimitado (depende del sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo)
Máximo número de índices por tabla	Ilimitado

Tabla 2.4 Limites de PostgreSQL

2.13.4 DBase

DataBase fue creado hace un cuarto de siglo, sus fundadores son Ashton-Tate. Desde 1999, debido a una inversión de millones de dólares se rediseñó lo que ahora es conocido como DBase Plus. DBase Plus es una plataforma avanzada de aplicaciones de bases de datos de desarrollo. Hoy en día la corporación DBase reporta que este gestor de datos es utilizado por millones de persona y en oficinas de gobierno por todo el mundo.

Entre sus características se encuentra la estabilidad, seguridad, soporte de ayuda, soporte de nuevos tipos de datos de campo (Unicode UTF-16), apoyo en la cancelación de operaciones, soporte de almacenamiento en caché, soporte XML, entre otras.**[50]**

2.13.5 MySQL

Es el sistema de gestión de bases de datos SQL Open Source más popular, es desarrollado, distribuido y soportado por MySQL AB.

Está disponible para la mayoría de las plataformas de sistemas operativos. Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema.
Las aplicaciones Apache-PHP-MySQL en conjunto son los más utilizados para la creación de páginas web.

Algunas de las características principales de MySQL: **[51]**

- Funciona en diferentes plataformas.
- APIs disponibles para C, C++, Java, Perl, PHP, Python.
- El servidor está disponible como un programa separado para usar en un entorno de red cliente/servidor. También está disponible como biblioteca y puede ser incrustado en aplicaciones autónomas. Dichas aplicaciones pueden usarse por sí mismas o en entornos donde no hay red disponible.

Son muchas las razones para escoger a Mysql como una solución de misión crítica para la administración de datos: **[52]**

- Costo: Mysql es gratuito para la mayor parte de los usos y su servicio de asistencia resulta económico.
- Asistencia: MysqlAB ofrece contratos de asistencia a precios razonables y existe una nutrida y activa comunidad Mysql.
- Velocidad: Mysql es mucho más rápido que la mayoría de sus rivales.
- Funcionalidad: Mysql dispone de muchas de las funciones que exigen los desarrolladores profesionales, como compatibilidad completa con ACID (Atomicity, Consistency, Isolation and Durability)

- Portabilidad: Mysql se ejecuta en la inmensa mayoría de sistemas operativos y, la mayor parte de los casos, los datos se pueden transferir de un sistema a otro sin dificultad.
- Facilidad de uso: Mysql resulta fácil de utilizar y de administrar. Las herramientas de Mysql son potentes y flexibles, sin sacrificar su capacidad de uso.

Límite	Valor
Máximo tamaño base de dato	Depende del tamaño de ficheros del SO
Máximo tamaño de tabla	65536 TB (256 ^ 7 - 1 bytes)
Número de registros	50 millones
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de índices por tabla	1000 bytes

Tabla 2.5 Limites de MySQL

El tamaño efectivo máximo para las bases de datos en MySQL usualmente los determinan los límites de tamaño de ficheros del sistema operativo, y no por límites internos de MySQL.

Sistema operativo	Tamaño máximo de fichero
Linux 2.2-Intel 32-bit	2GB (LFS: 4GB)
Linux 2.4	(usando sistema de ficheros ext3) 4TB
Solaris 9/10	16 TB
Win 32 w/FAT/FAT32	2GB/4GB
Win 32 w/ NTFS	2TB (posiblemente mayor)
MacOS X/ HFS +	2TB

Tabla 2.6 Tamaño de ficheros según SO

2.14 Tecnologías de Programación Web

En la actualidad existen distintas tecnologías que nos ayudan a crear páginas de web. A continuación se da una breve explicación de algunas de ellas.

2.14.1 ASP (Active Server Pages)

ASP es una herramienta poderosa para realizar páginas de internet dinámicas e interactivas. Esta tecnología fue desarrollada por Microsoft.

Debido a que es desarrollado por Microsoft utiliza un conjunto de servicios para servidores usando Microsoft Windows, este es llamado IIS (Internet Information Services).

Para ejecutar IIS se debe tener instalados Windows NT 4.0 o posterior, si se desea ejecutar una PWS (Personal Web Server), que es la versión de Microsoft de un programador de servidor Web para usuarios de PC que quieran compartir páginas Web y otros archivos de su disco duro, se debe tener Windows 95 o posterior.

Esta tecnología cuenta con otras dos, las cuales son ChiloASP e INstantASP, estas dos pueden funcionar y ejecutar respectivamente tecnología ASP sin Windows

Desde agosto de 2007 es el segundo sistema popular de servidor web funcionando en el 35% de los servidores de todos los sitios web.

Cuando un navegador solicita un archivo ASP, IIS pasa la solicitud al motor de ASP. Este motor lee el archivo ASP, línea por línea y ejecuta las secuencias de comandos en el archivo. Por último, el archivo ASP regresa al navegador como HTML plano.

Un archivo ASP puede contener texto, HTML, XML y secuencias de comandos. Las secuencias de comandos se ejecutan en el servidor. Los scripts en los archivos ASP son ejecutados en el servidor.

ASP puede:

- Dinámicamente editar, cambiar o añadir cualquier contenido de una página web.
- Responder a las consultas de los usuarios o de los datos presentando formularios HTML
- Acceso a los datos o bases de datos y mostrar los resultados en un navegador.
- Personalizar una página web para hacerla más útil a cada persona.
- Provee seguridad, debido a que el código ASP no se puede ver desde el navegador.
- Sencillez y velocidad.
- Puede minimizar el tráfico de red debido a la programación inteligente de ASP.

2.14.2 ASP. NET

ASP.NET es la próxima generación mejorada de ASP. Esta tecnología es parte de Microsoft, NET Framework y una poderosa herramienta para crear páginas dinámicas e interactivas.

ASP.NET es una tecnología nueva para el servidor de secuencias de comandos. Fue escrito desde cero y no es compatible con ASP clásico. Esta tecnología de script del lado del servidor permite que las secuencias de comandos.

Cuando un navegador solicita un archivo HTML, el servidor devuelve el archivo. Cuando un navegador solicita un archivo APS.NET, IIS para la solicitud al motor de ASP.NE , este lee el archivo,

línea por línea y ejecuta las secuencias de comandos, por último, el archivo de ASP.NET se devuelve al navegador como HTML plano.

.NET Framework es una infraestructura para la plataforma Microsoft .NET. este entorno sirve para la creación, implementación y ejecución de aplicaciones web y web services. **[52]**

2.14.3 PHP (Hypertext Preprocesor)

PHP es una poderosa tecnología de código abierto que nos permite crear páginas web dinámicas e interactivas. Aunque es la competencia de ASP de Microsoft, PHP es ampliamente utilizado debido a que es eficiente y libre, tanto para descargar y utilizar.

PHP es un servidor de lenguaje de scripts, como ASP, los scripts se ejecutan en el servidor. Esta tecnología soporta muchas bases de datos, entre ellas: MYSQL, INformix, Oracle, Sybase, Solid, PostgreSQL, etcétera.

Un archivo PHP puede contener texto, etiquetas HTML y scripts. Son devueltos al navegador como HTML plano. Los archivos pueden tener la extensión ".php", ".php3" o ".phtml".

La combinación PHP con MySQL es de las mejores debido a que son multiplataforma, es decir, se puede desarrollar en Windows y servir en una plataforma Unix.

MySQL es un gestor de base de datos que es ideal tanto para aplicaciones pequeñas y grandes, soporta el estándar. Una de las características más importantes de MySQL es que es gratuito para su descarga y utilización. **[52]**

Dentro de las características más importantes para desarrollar por medio de PHP están:

- PHP corre en diferentes plataformas. (Windows, Linux, Unix, etc.)
- Es compatible con casi todos los servidores utilizados en la actualidad. (Apache, IIS, etc.)
- Se puede descargar gratis a partir del recurso oficial de PHP. (www.php.net)
- Es fácil de aprender y se ejecuta de manera eficiente a lado del servidor.

2.14.4 JSP (Java Server Pages)

Con JSP podemos crear aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintáxis Java. **[53]**

El motor de páginas JSP está basado en servlets de Java, que son programas destinados a ejecutarse en el servidor.

En JSP se crean páginas de manera parecida a ASP o PHP. Se generan archivos con extensión .jsp que incluyen dentro de su código etiquetas HTML y las sentencias java a ejecutar en el servidor. El motor JSP lleva a cabo una traducción de esa página en un servlet, implementando un archivo class. **[54]**

2.15 Tecnologías de Diseño Web

2.15.1 JavaScript

Es un lenguaje de scripting diseñado para añadir interactividad a las páginas HTML. Es incrustado directamente en páginas HTML.

JavaScript es un lenguaje interpretado, lo que significa que los scripts se ejecutan sin recopilación preliminar. Todo el mundo puede utilizar JavaScript sin necesidad de una licencia.

Algunas características de JavaScript son: **[52]**

- JavaScript ofrece a los diseñadores una herramienta de programación HTML.
- Se puede utilizar para validar los datos de un formulario antes de ser sometido a un servidor.
- Puede ser detectado para detectar el navegador del usuario y dependiendo del mismo cargar otra página diseñada específicamente para ese navegador.
- Se puede utilizar para crear “cookies”.

2.15.2 XML (Extensible Markup Language)

XML fue diseñado para transportar datos, no para mostrarlos, está diseñado para ser auto-descriptivo.

XML y HTML fueron diseñados con objetivos diferentes.

- XML fue diseñado para transportar y almacenar datos, con especial atención a los datos.
- HTML se diseño para mostrar los datos, con énfasis en cómo se ve.

XML no es un reemplazo para HTML ya que en la mayoría de las aplicaciones web XML se utiliza para el transporte de datos, mientras que HTML se utiliza para formato y muestra de los datos. **[51]**

2.15.3 AJAX (Asyncrhronous JavaScript and XML)

- AJAX es una nueva técnica para la creación de una página web mejor, rápida e interactiva. No es un lenguaje de programación. **[55]**
- Esta tecnología se basa en los estándares de JavaScript, XML, HTML y CSS
- Con AJAX, un JavaScript puede comunicarse directamente con el servidor, con el objeto XMLHttpRequest.

- Utiliza transferencia de datos asincrónicos entre el navegador y el servidor web, permitiendo a las páginas web solicitar información por sección desde el servidor en lugar de toda la página entera. **[48]**

2.15.4 CSS (Cascading Stile Sheets)

Con CSS se puede controlar el texto (como fuente, color, tamaño, etc.), y el diseño (como fondos, margen, relleno, etc.) de un sitio web en un solo archivo.

Los estilos definen como mostrar los elementos HTML y XML, separando el contenido de la presentación. Permite a los desarrolladores web controlar el estilo y el formato de múltiples páginas web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afecta a todas las páginas vinculadas a esa CSS en las que aparezca. **[52]**

Separar la definición de los contenidos de una página web presenta ventajas, como:

- Crear documentos definidos
- Mejora la accesibilidad del documento
- Reduce la complejidad de su mantenimiento
- Permite visualizar el mismo documento en infinidad de dispositivos.

CAPÍTULO III

DESARROLLO

CAPÍTULO III DESARROLLO

3.1 Propuesta de solución

El Instituto de Ciencias Médicas y Nutrición Salvador Zubirán, cuenta con diferentes máquinas para realizar la sesiones de hemodiálisis; sin embargo, en la actualidad, las lecturas de los parámetros fisiológicos del paciente son tomadas por las enfermeras viendo el monitor de las máquinas, dicho proceso se repite constantemente durante una sesión de diálisis.

Como propuesta de solución al proceso que existe actualmente, se desea implementar una central de monitoreo la cual pueda obtener los parámetros de las máquinas de Hemodiálisis vía interfaz serial con el puerto que tienen las mismas máquina, estos parámetros serán procesados e interpretados por un Microcontrolador el cual tiene funcionalidad HTML y al mismo tiempo podrá generar una página web donde serán mostrados dichos parámetros, al mismo tiempo este microcontrolador se conecta mediante interfaz Ethernet hacia una red de computadoras con lo que facilita la visualización de dichos parámetros desde cualquier computadora conectada a la red, lo que sería la central de monitoreo como tal, por otro lado, una vez obtenidos estos parámetros en la interfaz web podrán ser almacenados en una base de datos la cual estará previamente cargada con los datos de cada paciente, lo cual automatizará el proceso de monitoreo y al mismo tiempo de la obtención de estadísticas de los parámetros fisiológicos, con la ventaja de que en una sola computadora se pueden estar monitoreando varias máquinas al mismo tiempo y esto puede ser desde cualquier lugar con acceso a la red donde esté conectada la máquina de hemodiálisis.

3. 2 Diagrama a Bloques

Para el desarrollo de este proyecto un primer paso fue identificar de forma general de qué elementos físicos se compondría, cuál sería su relación y la manera en la que se comunicarán. Para ello se diseño un diagrama a bloques y que se puede observar en la figura 3.1.

3.2.1 Tarjeta de Desarrollo Microchip PICDEM NET 2

La tarjeta de desarrollo PICDEM.net.2 es una tarjeta de Internet/Ethernet capaz de soportar el controlador de Ethernet ENC20J60 y el chip de la familia de los microcontroladores Ethernet, el PIC18F9J60, la cual fue la principal razón de su utilización; cabe mencionar que aunque se describe todos los elementos que contiene la tarjeta, solo fueron útiles algunos de ellos para la realización de dicho proyecto.

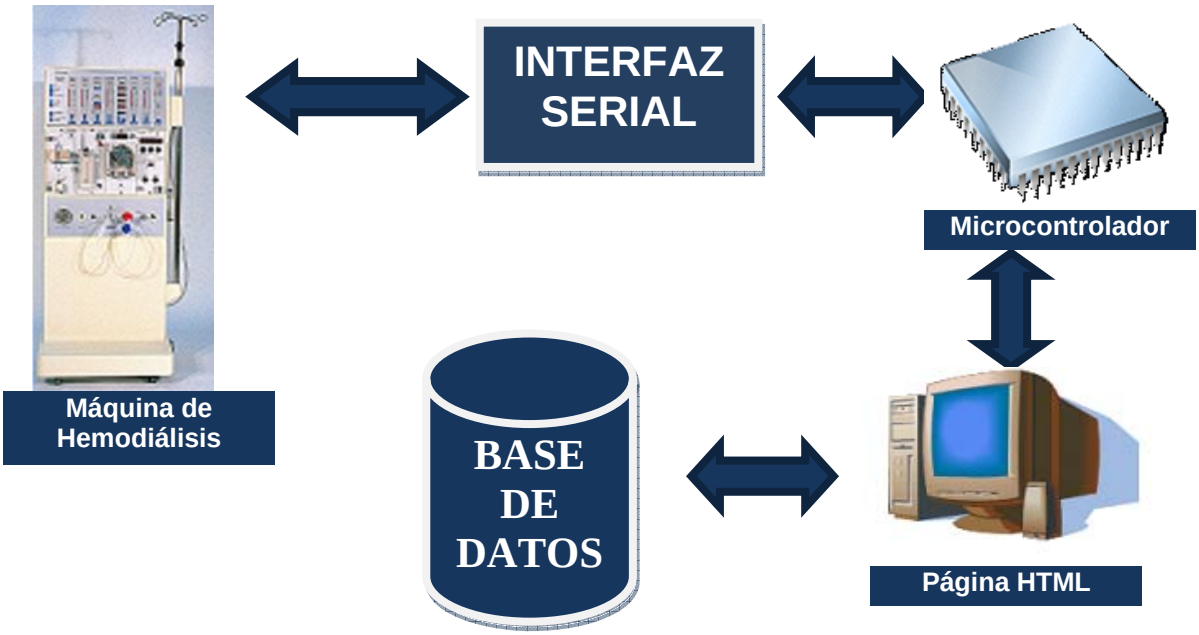


Figura 3.1 Diagrama a bloques del prototipo del proyecto

La tarjeta cuenta con características que nos son útiles para realizar nuestra aplicación

- Servidor Web con soporte HTML
- Soporte de Microchip para la pila TCP/IP
- Interfaz Ethernet (conectores RJ-45)
- Display de 16x2 caracteres alfa-numéricos
- Botones y LEDs Programables
- Interfaz RS-232

En la figura 3.3 se describen cada una de las características de la tarjeta de desarrollo **[58]** las cuales se describen a continuación:

1. MICROCONTROLADOR El microcontrolador PIC18F97J60 que viene junto con el controlador Ethernet. Dicho dispositivo esta sincronizado a 25 MHZ y puede programarse el firmware usando el stack TCP/IP de Microchip.

2. CONTROLADOR ETHERNET Este dispositivo provee de una conexión a Ethernet al microcontrolador utilizando una interfaz SPI. (Serial de Interfaz Periférico)

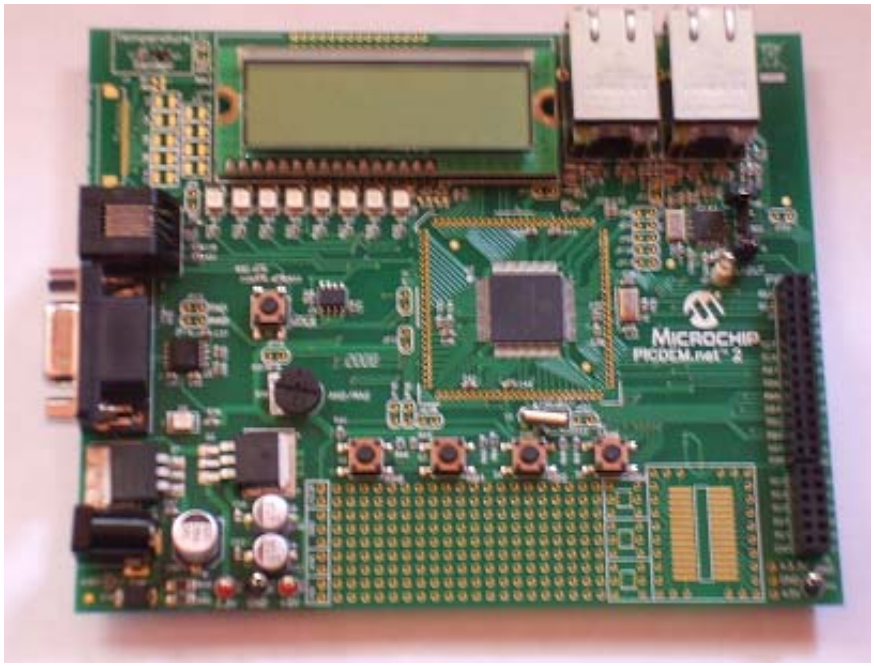


Figura 3.2 Tarjeta de Desarrollo Microchip PICDEM NET 2

3. MEMORIA. La memoria EEPROM proporciona 256 Kbits (32 Kbytes) de almacenamiento para páginas web y no volátil para opciones de configuración, es programable por medio de una interfaz SPI (Serial de Interfaz Periférico)

4. PANTALLA LCD. Con dos líneas de 16 caracteres.

6. Sensor de Temperatura. Es un sensor de temperatura analógico, modelo TC1047, el esta conectado a un pin de I/O analógica del microcontrolador.

7. LEDS DEFINIDOS POR EL USUARIO. Ocho LEDs son controlados por puertos de I/O digital del microcontrolador (PORTJ) y pueden ser utilizados para simular una salida digital para controlar algún dispositivo. Estos también pueden ser habilitados o deshabilitados.
8. ÁREA DE BOTONES DEFINIDA POR EL USUARIO. Estos switches son conectados a puertos de I/O digital del microcontrolador (PORTB) y pueden ser utilizados para simular una entrada digital.

9. POTENCIÓMETRO DEFINIDO POR EL USUARIO. El potenciómetro de 1 a 10 KOhm es conectado a un pin de I/O analógica del microcontrolador.

10. BOTON PARA REINICIAR Este interruptor está ligado al pin MCLR del microcontrolador y es usado para reiniciar la tarjeta.

11. RJ-45 (10Base-T) CONECTORES MODULARES. La tarjeta esta equipa con dos módulos de conector, uno para el PIC18F97J60 y otro para el ENC28J60.

12. RJ-11 CONECTOR MODULAR. Permite a la tarjeta de desarrollo conectarse con el Entorno de Desarrollo de Microchip MPLAB® por medio de programador universal PICKIT 2® para la programación del microcontrolador contenido en la tarjeta.

13. PUERTO SERIAL. La tarjeta incluye el puerto con un conector DB9. Esto permite la configuración de tarjetas IP y direcciones Ethernet mediante una norma de conexión serial. Este puerto también permite a los usuarios la descarga de nuevas páginas web a la EEPROM Serial con una capacidad de 256 Kb y que utiliza protocolo I2C.

14. PUERTOS I/O. Un par de header (J5 y J6), permiten el acceso directo a cinco de los puertos de I/O del microcontrolador.

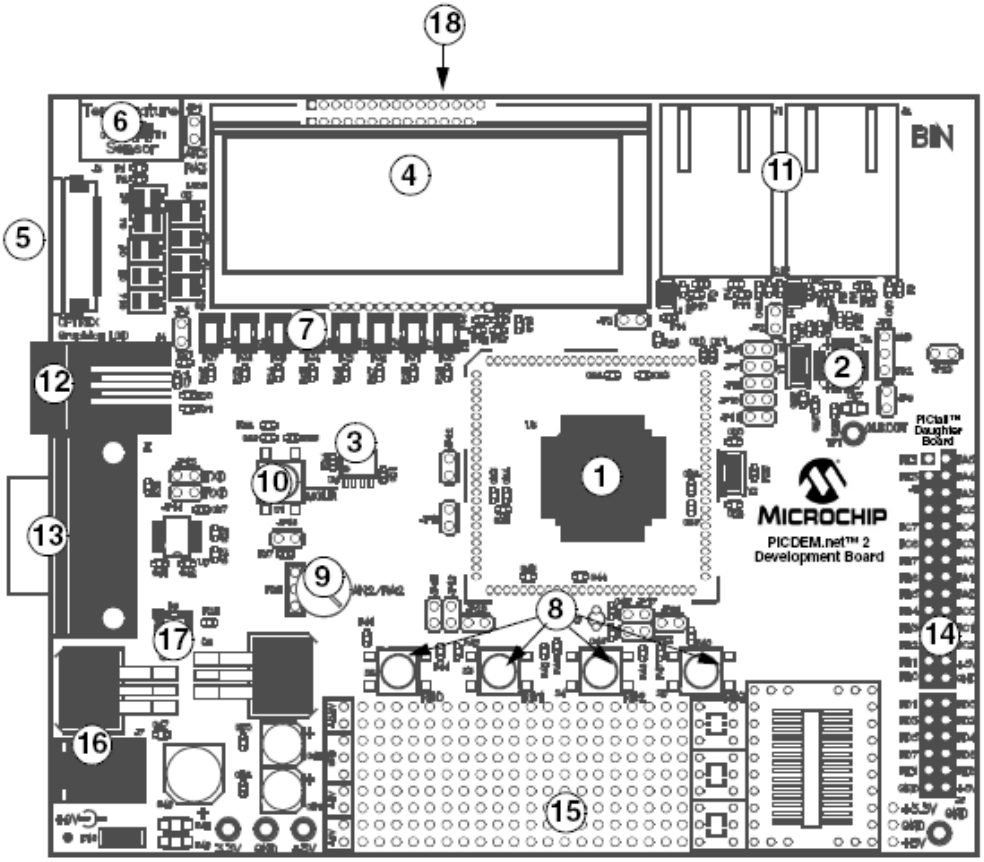


Figura 3.3 Características que incluye la Tarjeta de Desarrollo Microchip PICDEM NET 2

16. VOLTAJE DE ALIMENTACIÓN. La tarjeta utiliza 5 y 3.3 VDC, tiene una entrada en J7 de 9VDC a 500 mA y dos reguladores independientes para los voltajes de 5 y 3.3 ya mencionados.

17. INTERFASES ETHERNET. Cuenta con dos canales de transmisión independientes a través de una única dirección MAC (Control de Acceso al Medio). La cual es usada para la trasmisión Ethernet, para la identificación y filtrado de paquetes.

En el Capítulo II se realizó una comparativa sobre las diferentes familias de los dispositivos necesarios, así como una descripción de sus características por lo que a continuación se mencionan los componentes necesarios para el desarrollo de este proyecto teniendo como base el análisis anterior.

3.2.2 Microcontrolador Microchip PIC18F97J60

La función principal de este componente es almacenar en su memoria una página HTML cuyo contenido son los datos tratados, provenientes de la Máquina de hemodiálisis y a la vez reflejados a la máquina donde de donde se hace la petición de dichos datos.

Se utilizó para el desarrollo del proyecto el modelo PIC18F97J60 de la marca Microchip Technology Inc., de la familia PIC de la serie 18F porque está optimizado para aplicaciones embebidas y además nos ofrece un Controlador Ethernet 10BASE-T que ya viene integrado.

La decisión final de utilizar este microcontrolador también se basa en que nos ofrece una pila de software TCP/IP optimizada para los PIC18 gratuita y editable cuyo compilador es amigable y familiar para las integrantes de dicho proyecto.

La tabla 3.1 muestra más características técnicas más importantes y dominantes de los microncontroladores PIC18F97J60

3.2.3 Controlador Ethernet Microchip ENC28J60

La función principal de este dispositivo independiente que posee interfaz SPI (Serial Peripheral Interface Bus) ó Serial de Interfaz Periférico, es conectar al microcontrolador a una red Ethernet y proporcionar de un módulo interno de acceso directo a la memoria para mayor rendimiento del procesado de datos (ver figura 3.5).

Características del dispositivo	**PIC18F97J60**
Frecuencia de Operación	41.667 MHz
Memoria de Programa (KBytes)	96
Memoria de Programa (Instrucciones)	49,148
Memoria de Datos (Bytes)	3808
Fuente de Interrupción	29
Puertos de I/O	A,B,C,D,E,F,G,H,J
Pines de I/O	70 repartidas en 9 puertos
Timers	5
Módulos de captura, comparación y modulación por ancho de pulsos (CCP)	2
Módulos mejorados de captura, comparación y modulación por ancho de pulsos (ECCP)	3
Comunicación Serial	MSSP (2), USART (2)
Comunicación Ethernet (10Base-T)	SI
Puerto de Comunicación Paralelo Esclavo (PSP)	SI
Bus de Memoria Externa	SI
Modulo Analógico-Digital 10 Bit	16 Canales de entrada
Reset (y Retardo)	POR, BOR, MCLR, WDT (PWRT, OST)
Set de Instrucciones	75 Instrucciones, 83 con extensión opcional de Instrucciones
Empaquetado Fisico	100 pines

Tabla 3.1 Características del microcontrolador PIC18F97J60

El controlador ENC28J60 consiste en los siguientes bloques funcionales

1. Una interfaz SPI que sirve como un canal de comunicación entre el microcontrolador y el ENC28J60

2. Una memoria RAM de puerto dual para la recepción y transmisión de paquetes de datos.

3. El bus de interfaz que interpreta datos y órdenes recibidas la Interfaz SPI

4. El modulo MAC (Control de Acceso al Medio) que implementa IEEE 802.3

5. El PHY (Capa Física) el módulo que codifica y descifra los datos análogos que están presentes sobre el interfaz de par torcido.

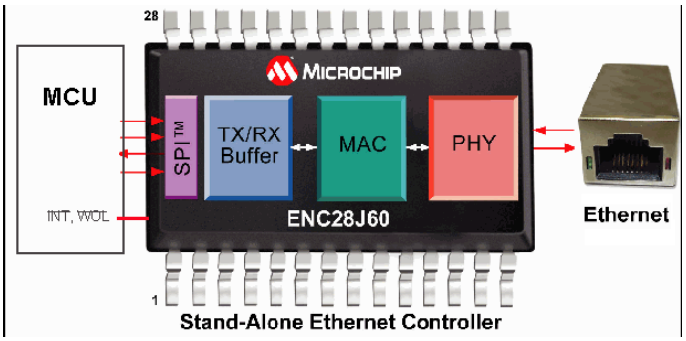


Figura 3.5 Controlador Ethernet Microchip ENC28J60

3.2.4 Manejador de Puerto Serial MAX3232C

Es un circuito integrado que consiste en una carga doble regulada que proporciona voltajes de salida de +5.5V, independientemente del voltaje de entrada, de los 3 – 5.5V. La carga opera in un modo discontinuo, si el voltaje de salida es menor que 5.5V, la carga es habilitada y si el voltaje de salida excede de 5.5V, la carga es deshabilitada.

El MAX3232 es un circuito integrado que convierte los niveles de las líneas de un puerto serie RS232 a niveles TTL o CMOS y viceversa. El MAX3232 soluciona la conexión necesaria para lograr comunicación entre el puerto serie de una PC y cualquier otro circuito con funcionamiento en base de señales e nivel TTL/CMOS.

Estos convertidores son suficientes para manejar las cuatro señales mas utilizadas del puerto serie que son TX, RX, RTS y CTS.

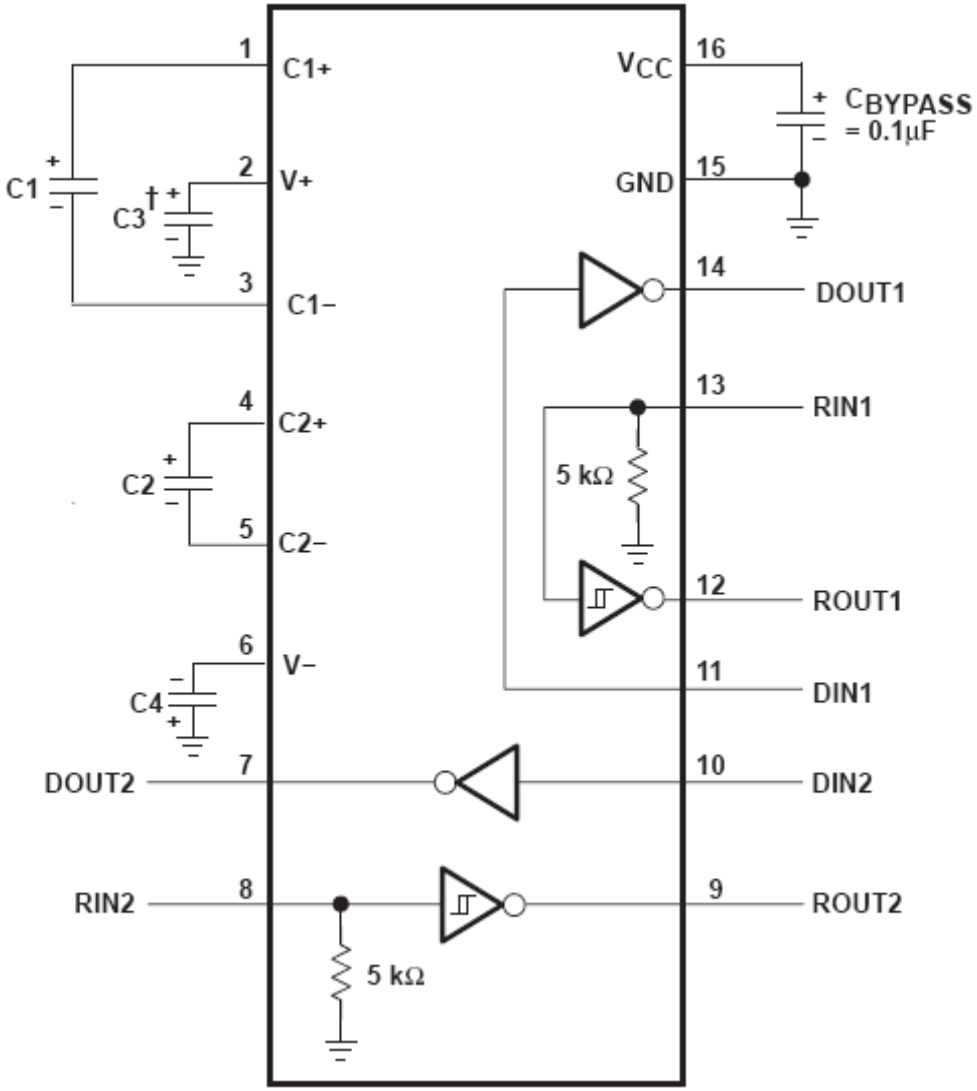


Figura 3.6 Controlador Serial MAX3232C Texas Instruments

3.3 Lenguaje de Programación Microchip C18

La elección del lenguaje de programación no hubo duda, ya que en este tipo de microcontroladores el código se realiza en lenguaje C.

MPLAB® C18 es un compilador de C e incluye el compilador, y las librerías necesarias para cada tipo de microcontrolador y al igual que un ensamblador, el compilador C18, traduce de código de programación a lenguaje máquina.

Para dicho proyecto se utilizó la versión académica (demo de 60 días) que nos proporciona también Microchip **[60]**

Cuenta con las siguientes Características:

- C18 toma el estándar de lenguaje de programación C y es escrito utilizando la notación del estándar ANSI C
- El compilador puede optimizar código utilizando rutinas que fueron empleadas en una función para ser usadas por otras funciones.
- Es capaz de compartir fragmentos de código entre varias funciones.

3.4 Microchip Stack TCP/IP

La empresa Microchip desarrollo una pila de software TCP/IP, para el desarrollo de aplicaciones basadas en microcontrolador y Ethernet, el cual lo dejo libre para el desarrollo de otras aplicaciones. Este software se utilizó como base para el desarrollo de nuestro proyecto, el cual sufrió algunas modificaciones

Basada en el modelo de referencia TCP/IP, la pila se divide en múltiples capas, la misión de cada una de las capas es proveer servicios a las capas superiores, haciéndoles transparentes el modo en que esos servicios se llevan a cabo. **[57]**

La pila está escrita en lenguaje de programación C. En conjunto con el controlador Ethernet, permite reducir los tiempos y costes de desarrollo de aplicaciones de una red de área local o internet

La pila de TCP/IP de Microchip, incluye las siguientes características principales

- Optimizado para todas las familias PIC18, PIC24, PIC32
- Soporta los protocolo: ARP, IP, ICMP, UDP, TCP, DHCIP, SNMP, HTTP, FTP, TFTP
- Dispositivos Ethernet
- Es compatible con los compiladores MPLAB C18, C30 Y C32.

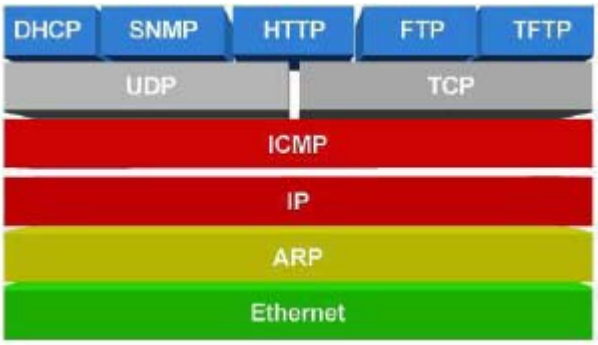


*Figura 3.7 Capas TCP/**IP***

3.5 Herramientas de Desarrollo MPLAB

MPLAB, es un programa que sirve para desarrollar aplicaciones de microcontrontroladores Microchip, es llamada Ambiente de Desarrollo (IDE), porque provee de un ambiente que contiene todos los elementos necesarios (Gestor de Proyectos, Editor de textos y Barra de Herramientas), además de que permite Escribir y Compilar, todo esto para desarrollar y desplegar aplicaciones para sistemas embebidos.

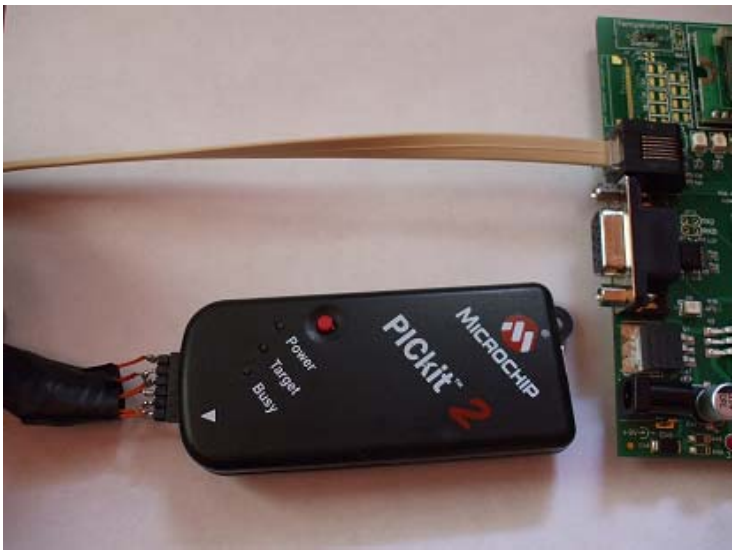
MPLAB versión 8,3 es el software que se utilizó para este proyecto y es un software gratuito que nos proporciona Microchip. **[58]**

3.6 Programador PIC KIT2

Para el desarrollo de dicho proyecto utilizamos el programador PIC KIT2 que nos permite la funcionalidad de depuración y programación del Microcontrolador y ofrece las siguientes características:

- Programa PICS desde la familia 10F, PIC13F, PIC14F, PIC16F, PIC18F, PIC24F, dsPIC30, y dsPIC33
- Comunicación USB.
- Trabaja con WINDOWS XP y WINDOWS VISTA.
- Puede programar los microncontroladores montados directamente en su aplicación y/o protoboard.
- Facilidad y rapidez en su uso.
- Incluye CD de programa, manual de instalación, cables de conexión.

En el conector **RJ-11** que viene en la Tarjeta de Desarrollo se encuentran los pines de programación, los cuales se conectan a través de un cable cuyas salidas se muestra a continuación, y este cable de interfaz modular a su vez es conectado al Programador PicKit2. **[59]**



Pin Description

1 = VPP/MCLR
2 = VDD Target
3 = Vss (ground)
4 = ICSPDAT/PGD
5 = ICSPCLK/PGC
6 = Auxillary

Figura 3.8 Programador PIC KIT2

3.7 Microchip File System (MPFS)

Es un sistema de archivo desarrollado por Microchip, que se usa en el servidor HTTP Interno para guardar páginas Web. MPFS sigue un formato especial para guardar múltiples archivos en una sola imagen.

Microchip proporciona esta aplicación dentro de la Microchip Stack TCP/IP, para nuestro caso se utilizo la versión 1.0 que se ejecuta en línea de comando y es de la que se hablo previamente, sin embargo a partir de la versión 5.0 de la Microchip Stack TCP/IP esta disponible la versión 2.0 de la aplicación MPFS (ver figura 3.9), la cual ya funciona en un entorno de ventanas, además esta versión ofrece la versión de grabar la imagen MPFS directamente en la memoria externa de la tarjeta de desarrollo sin necesidad de modificar el código o recompilar la aplicación, para ello se hace uso del protocolo FTP incluido en la Microchip Stack TCP/IP y ya programado en la tarjeta de desarrollo, lo anterior ofrece grandes ventajas, dado que se puede modificar la página web sin necesidad de modificar la funcionalidad propia de la tarjeta, para el propósito del presente proyecto únicamente se utilizo la versión 1.0 de la utilería MPFS debido a que no se necesitaba utilizar el protocolo FTP.

3.8 Aplicación en C18

3.8.1 Comunicación serial con el microcontrolador

Antes de comprender la parte de la comunicación serial, es necesario familiarizarse con los componentes que participan, para ello se fueron creando diseños sencillos de programas que nos facilitaron el desarrollo del diseño actual.

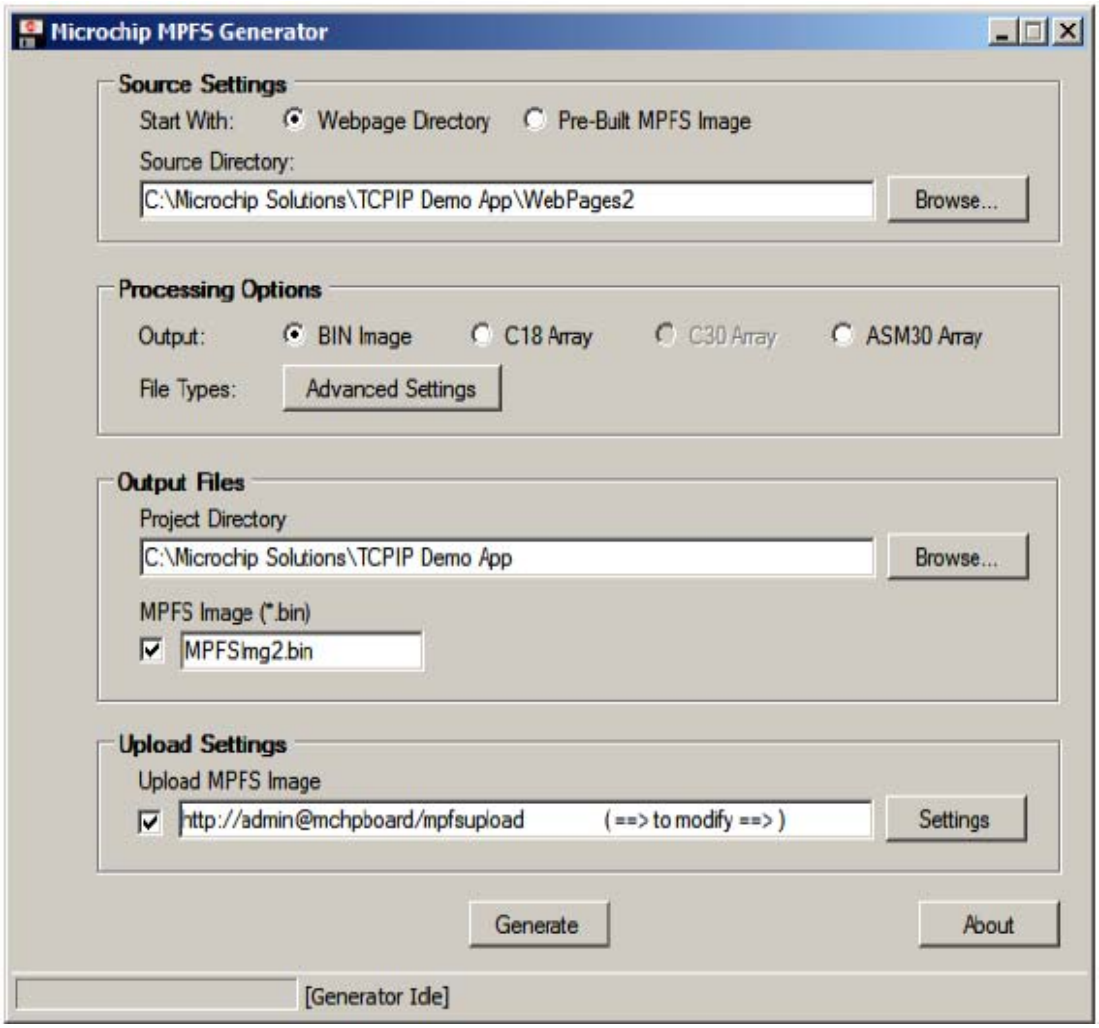


Figura 3.9 Herramienta MPFS 2.0

3.8.2 Aplicación Inicial

Este primer programa servirá para adentrándonos en el conocimiento y manejo de la tarjeta, principalmente del microcontrolador, del manejo de sus puertos de entrada y salida, haciéndolo evidente a través del encendido y apagado de led´s.

A continuación se presenta el algoritmo que contiene básicamente los pasos que se desarrollaron para la realización de dicho programa:

- Una vez que incluimos las librerías que se van a usar en el proyecto como la librería del PIC (*#include <PIC18FJ60.h>*), y la librería para el uso de retardos (*delay.h*)
- En seguida dentro de la función principal “main” declaramos nuestras variables, para este programa ocupando como puerto de Entrada el puerto B (PORTB) y como Salida el puerto J (PORTJ), las cuales se inicializan en cero.

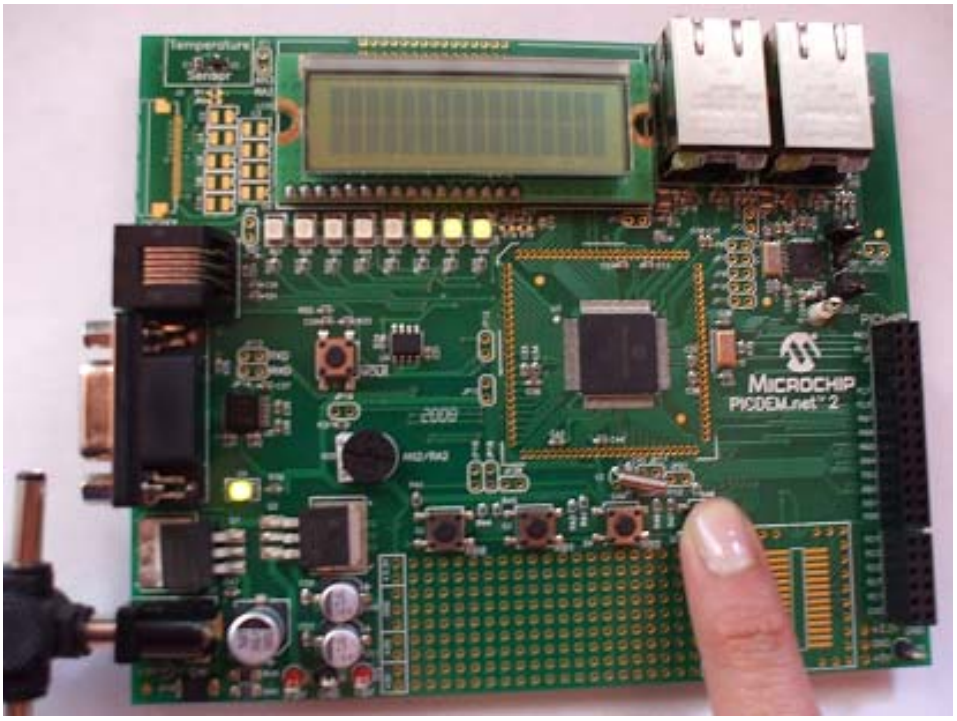


Figura 3.10 Primer prueba con la tarjeta-Encendido de led´s

- Y por último hacemos uso del ciclo recursivo *while* con un retardo entre cada parpadeo de 3(s).
- Es importante recalcar que para todos los programas desarrollados, usando el PIC18F97J60 cambian los bits de configuración ***Menu/Configure/Configuration Bits…***

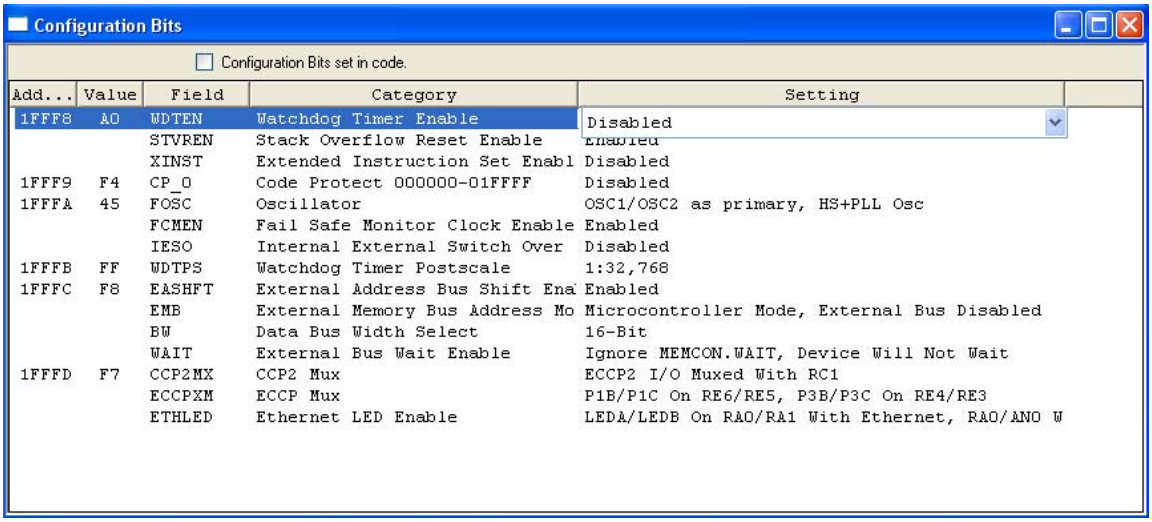


Figura 3.11 Configuración de bits en la IDE de MPLAB

3.8.3 Funcionamiento y Control del LCD

Una vez que se tuvo conocimiento sobre el microcontrolador se fueron creando varios programas para ir conociendo el funcionamiento de otros dispositivos como el LCD.

Este dispositivo solo trabaja en modo texto y despliega 32 caracteres, en 2 filas con 16 columnas y el objetivo principal de este segundo programa fue desplegar texto con el fin de dominar el funcionamiento y que posteriormente nos ayudara a verificar si recibimos datos de la máquina de Hemodiálisis vía puerto serial, desplegándose en la pantalla.

Figura 3.12 Segunda prueba con la tarjeta – Control LCD

Para el desarrollo de este programa se hizo uso de diferentes funciones que ejecutan tareas específicas que hacen que la aplicación funcione correctamente.
Estas funciones se explican a continuación

- **LCDdata:** Se encarga de escribir y validar los datos que se reciben
- **LCDinit:** Inicializa los comandos del LCD (Si el operaciones se realizan en 8 o 4 bits, Ubicación del cursor, si esta activo, borrado, etc.)
- **Busylcd:** Regresa el estado del LCD mientras realiza la lectura del LCD
- **stringtoLCD:** Se encarga del incremento de dirección del buffer

Para cualquier duda el programa completo lo pueden observar en la parte de Anexos.

Para el desarrollo del programa del funcionamiento del LCD tuvimos algunas dificultades pues la librería que viene por default en C18 para el LCD (xlcd.h ubicada en C:\mcc18\h\) no funciono para el Microcontrolador que estamos utilizando, ya que al compilar el programa nos marcaba errores de

funciones que no están presentes dentro del programa y que se encontraban dentro de dicha librería, es por esto que se tuvieron que reescribir.

La librería se reescribió bajo un esquema de:

- Las operaciones se realizan en un bus de 8 bits
- Cursor activo
- Borrado del display
- Cursor parpadeante
- Modo de entrada: incremento

3.8.4 Comunicación con el puerto Serie

Una vez que se comprendió el funcionamiento de la pantalla de LCD y del microcontrolador se prosiguió a hacer el desarrollo del programa que nos permitiera tener comunicación entre el puerto serie y el PIC.

Recordemos que uno de los propósitos del proyecto es obtener los datos provenientes de la máquina de Hemodiálisis vía puerto serie para transmitirlos en forma de una cadena de caracteres al microcontrolador y poder observarla a través del LCD y de esta forma verificar que se están enviando los datos.

|Para esta parte hicimos uso del software que nos proporciona Windows llamado Hyperterminal que nos sirvió de ayuda para mandar una cadena de caracteres a través del puerto serie de una computadora hacia el PIC y posteriormente la cadena escrita en la Hyperterminal se ve reflejado en la pantalla del LCD.

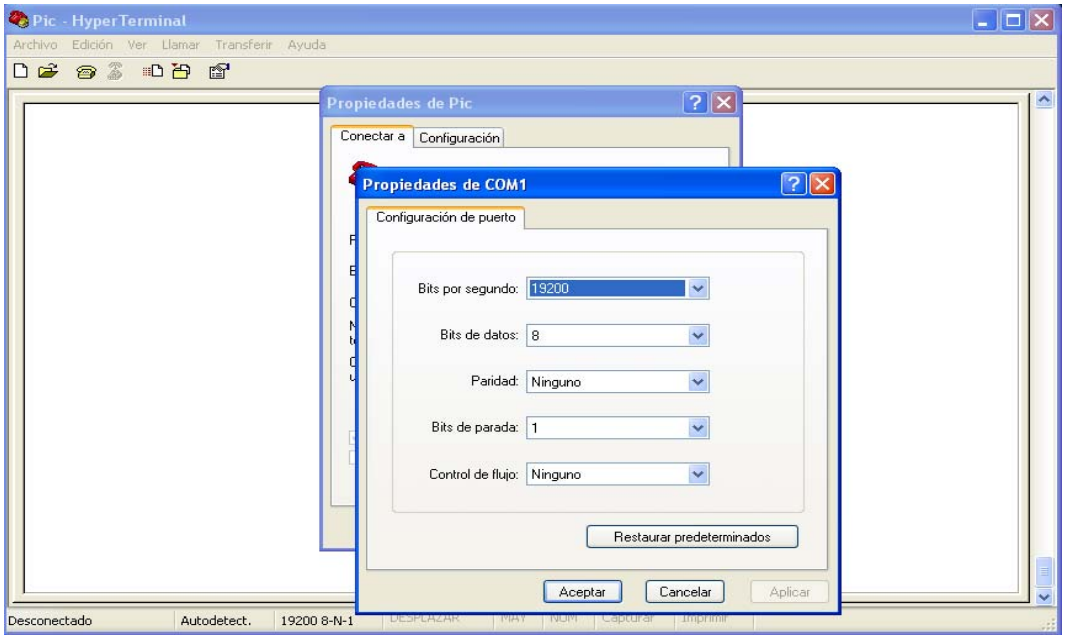


Figura 3.13 Configuración de puerto RS232 en la Hyperterminal

Como se pudo ver en el capítulo II nuestro microcontrolador está dotado de dos puertas de comunicación, para este programa utilizamos el puerto USART del microcontrolador que es una adaptador de comunicación serie síncrona y asíncrona y que adelante se explicará con más detalle.

Como primer paso se tuvo que hacer la configuración de las propiedades del puerto de comunicaciones, pues como sabemos todo tipo de comunicación serial tiene que realizarse en los tiempos correctos los cuales se deben establecer, debido a lo anterior los dispositivos que se comunican deben tener la misma velocidad (bps), en nuestro caso es de 19200, esta información se obtiene de las propiedades de la tarjeta de desarrollo, los demás parámetros quedan como se observa en la figura 3.14.

.

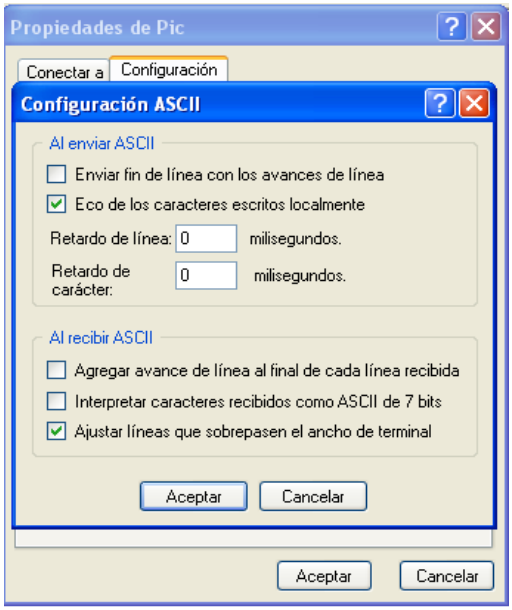


Figura 3.14 Configuración ASCII en desde la Hyperterminal

También es importante mencionar que antes de hacer la conexión, se debe configurar dentro de las propiedades, marcando que se envié eco de los caracteres escritos locamente.

Una vez que se abre la pantalla del Hyperterminal con el cursor parpadeando, al presionar el botón **RB2**, la terminar responde enviando un texto que pide una cadena, y una vez que se da la cadena y se presiona enter en la terminal, la cadena es enviada y mostrada al LCD.

La cadena que se muestra en LCD se guarda y con el botón **RB1** se muestra en la pantalla de la Hyperterminal nuevamente.

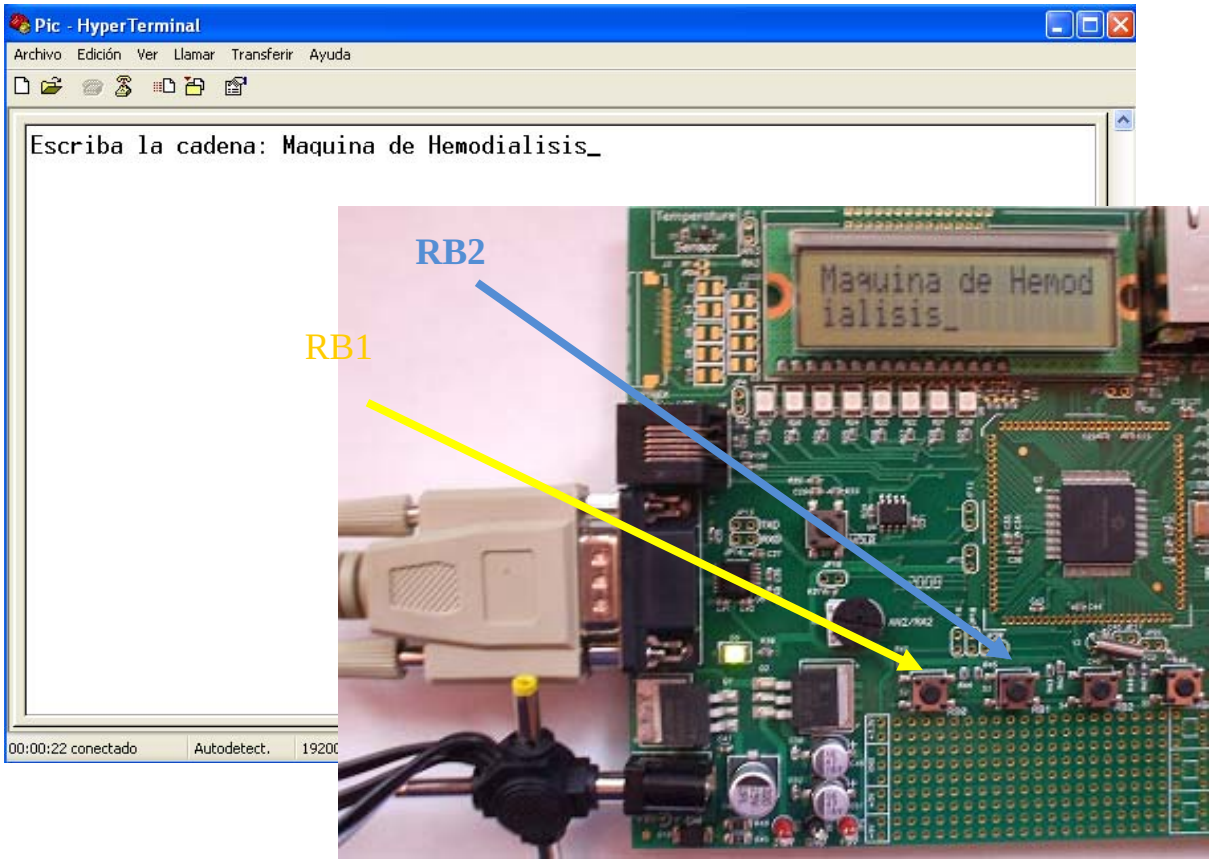


Figura 3.15 Tercer prueba con la tarjeta – Comunicación con el puerto serie

Como habiamos mencionado para la comunicación serial es necesario agregar la librería usart. Con esta librería se configura el modo de transmisión y recepción serie de nuestro microcontrolador. A continuación se hace una breve descripción de las funciones que participan dentro la librería USART, la cual contiene la configuración del modulo USART, si se tiene alguna duda se puede consultar los anexos, ahí encontrara el programa completo de dicha librería.

putrsUSART: Espera mientras el puerto esta ocupado, y una vez que se desocupa escribe una cadena de caracteres en la memoria del programa.

putcUSART: Se encarga de enviar un byte al USART.

BusyUSART: Determina si se ha transmitido el dato, es decir si esta transmitiendo o esta ocupado.

getcUSART: Lee un byte del puerto USART.

getsUSART: Lee una cadena de caracteres y las guarda en una cadena en la memoria de datos del puerto USART.

DataRdyUSART: Detecta si hay información lista para efectuar en una lectura del buffer de recepción.

putsUSART: Escribe una cadena de caracteres en la memoria de datos.

ReadStringUSART: Lee un carácter del buffer de recpción del puerto USART.

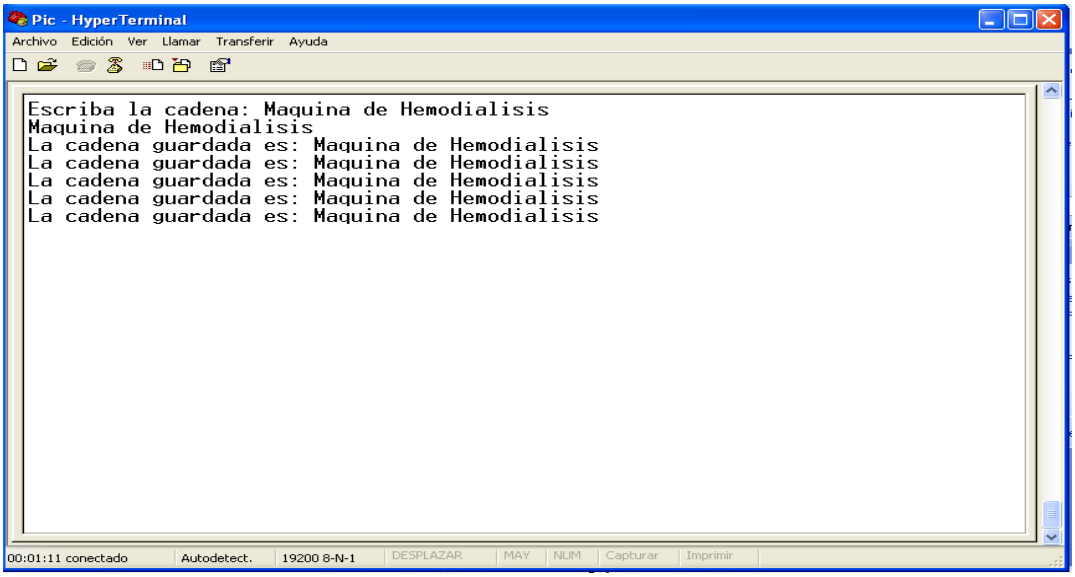


Figura 3.16 Comunicación con el puerto serie a través del Hyperterminal

En la Figura 3.17 se puede ver el diagrama de flujo que se requrió para la realización del programa

3.8.5 Generación de la página dentro del microcontrolador

Para llevar a cabo el diseño de la página web en el Microcontrolador se hizo uso de la herramienta MPFS, el proposito principal es empaquetar páginas web en un formato para el almacenamiento en un sistema embebido. Cuando se desarrolla una aplicación como la nuestra que usa el almacenaiento en la memoria del microcontrolador, la utilidad de la herramienta nos fue de gran ayuda para cargar imágenes a la memoria del microcontrolador.

MPFS sigue un formato especial para guardar archivos en una sola imagen, los formatos que salida pueden tener las extensiones “.c” ó “.bin”

Para la creación de una imagen MPFS hay que seguir los siguentes pasos :

1. Una ves que seleccionamos “Ejecutar” y tecleamos el comando “cdm” en la ventana de incio de Windows.

2. Tecleamos el directorio en donde se encuentra nuestra herramienta en nuestro caso es D:\mpfs.exe.

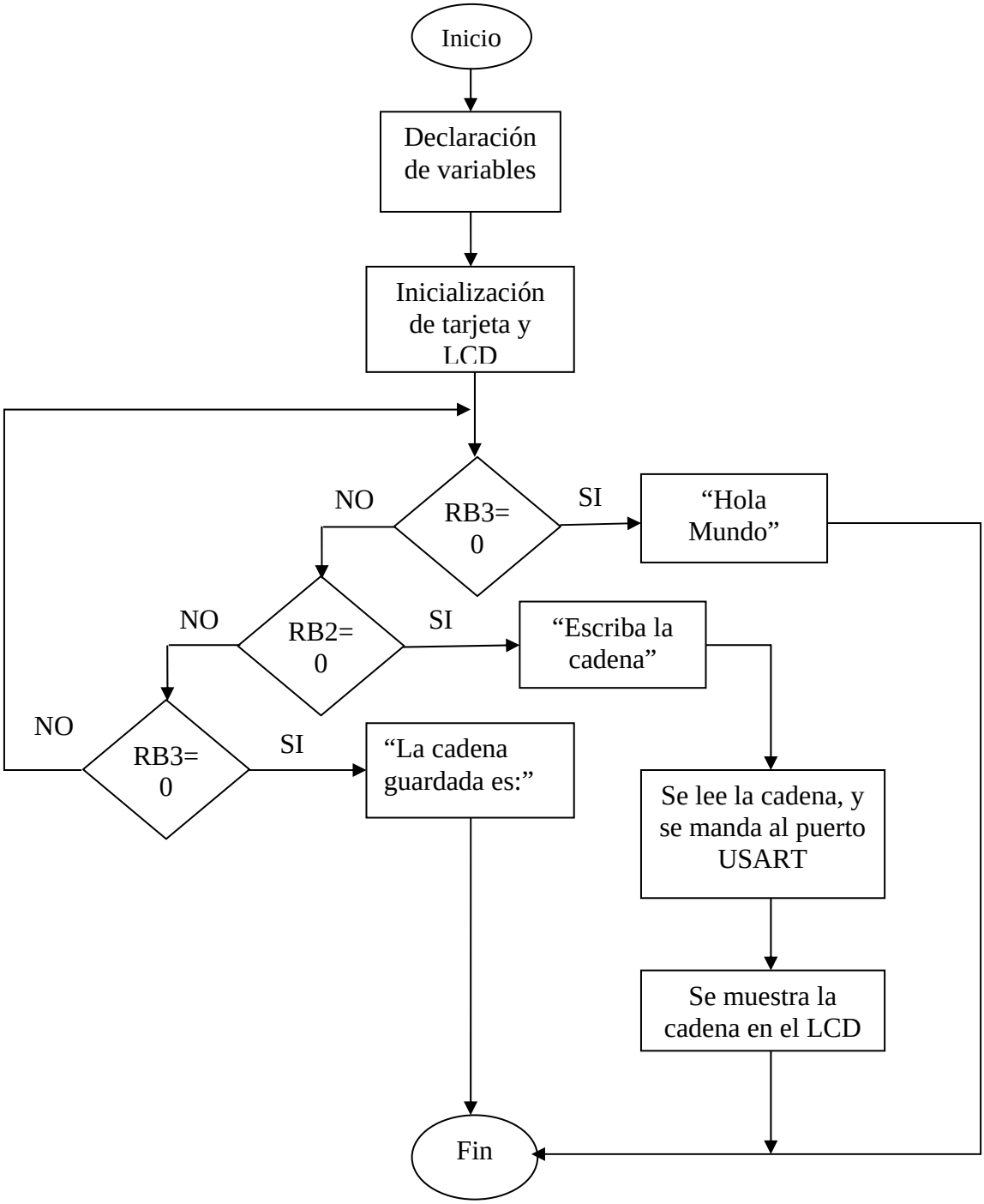


Figura 3.17 Diagrama de flujo del programa de comunicación con el puerto serie

3. Es importante mencionar que antes de crear la imagen se debe tener ubicado un mismo directorio donde se almacenaran todas las páginas creadas, ademas es necesario que el nombre de todos los archivos respeten el antiguo formato de DOS que es de 8 caractes de nombre seguido por un punto y 3 caracteres con la extensión del archivo. En nuestro caso nombramos a una carpeta "mypages" en el directorio D:\MCHPTCPStack 3.75\MPFS.exe.

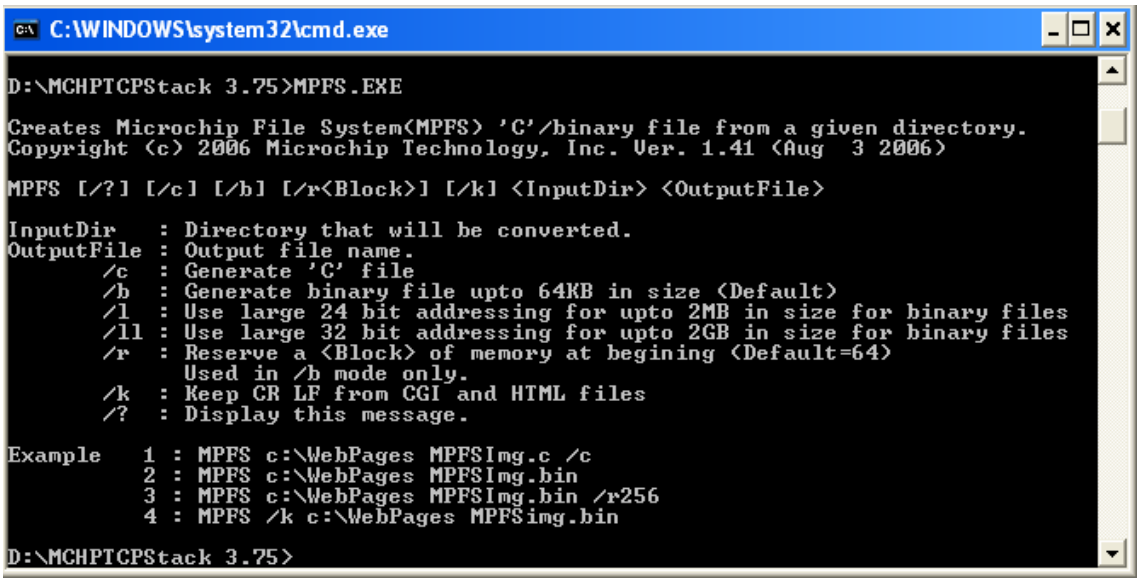


Figura 3.18 Comandos de la utilería MPFS

4. Posteriormente se crea la imagen con la siguiente sintaxis en la línea de comandos.
   MPFS d:\mypages MPFSImg.c \c de esta forma se genera un archivo de datos "c" contenido en el directorio C:\mypages que es el directorio donde se contienen las páginas web.
5. El archivo creado se puede observar dentro del proyecto en MPLAB y se muestra en la figura 3.19

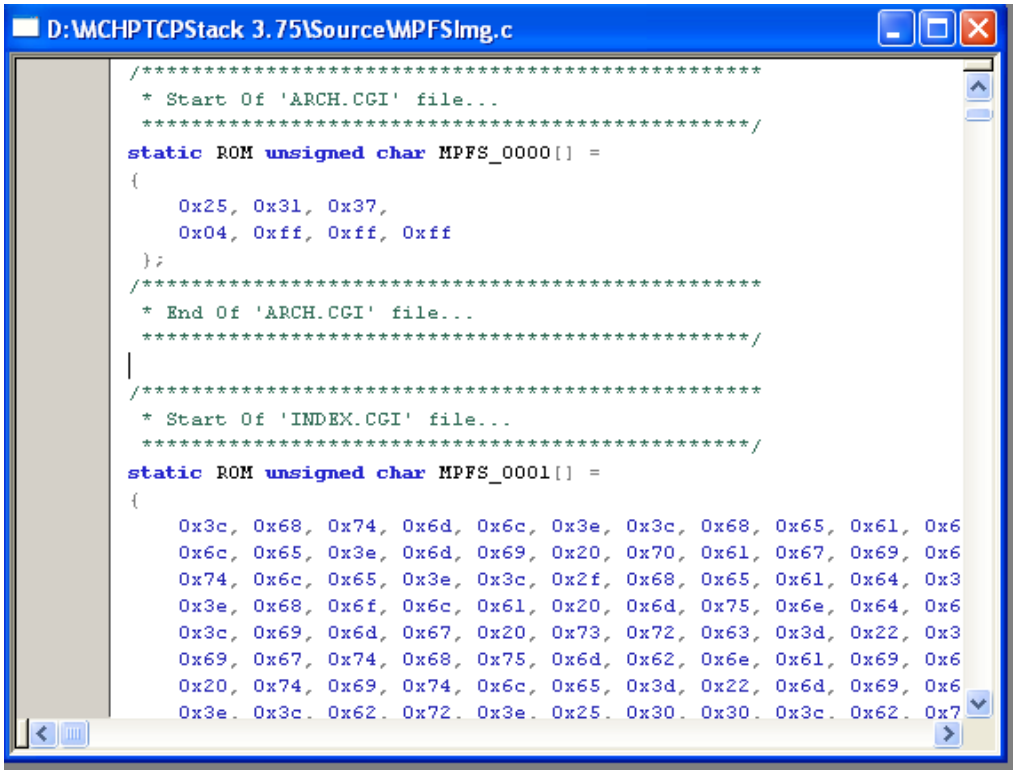


Figura 3.19 Archivo generado por la herramienta MPFS

3.8.6 Comunicación Ethernet

Como ya se explico anteriormente la tarjeta de desarrollo cuenta con el servicio DHCP dentro del software preinstalado (Microchip Stack TCP/IP), sin embargo para poder configurar de esta manera, es necesario el uso de un ruteador que sea capaz de asignar una IP o que cuente con el servicio de DHCP, tal como se muestra en el siguiente esquema.

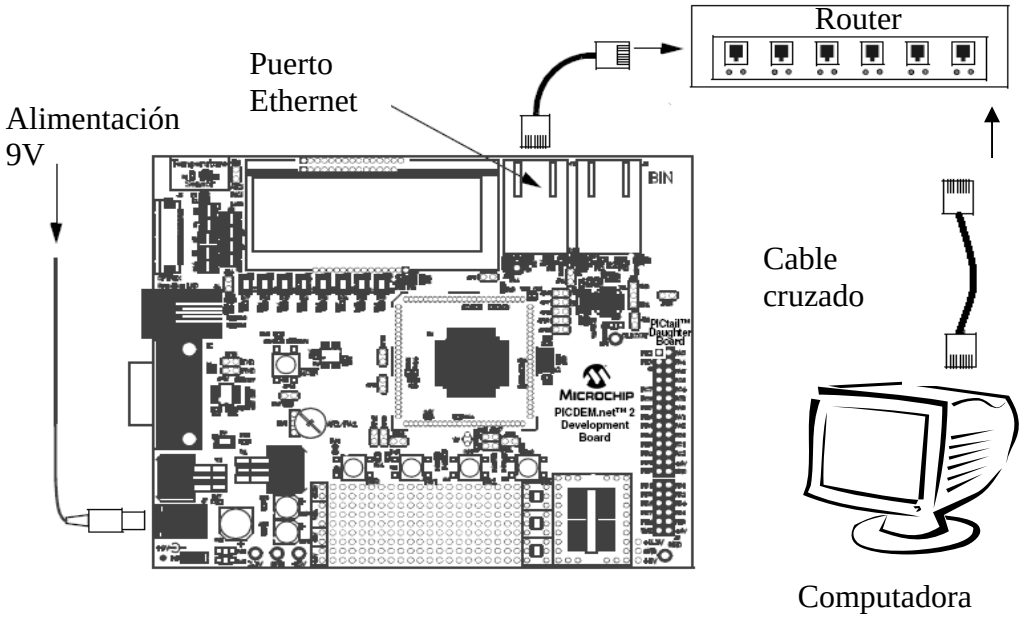


Figura 3.20 Conexión de la tarjeta de desarrollo usando DHCP

Una vez que la tarjeta de desarrollo es conectada al ruteador y este detecta el servicio de DCHP, la dirección IP aparecerá en el LCD de dicha tarjeta, quedando configurada correctarme.

Por otro lado es posible también configurar la tarjeta para que funcione de manera directa mediante una conexión punto a punto, para ello se requiere de un cable de conexión Ethernet cruzado Figura 3.21.

También es necesario editar el codigo fuente en la funcion *InitAppConfig* dentro del archivo MainDemo.c, para esto se le asigna un valor constante a la variable *response,* la cual debe de ser una cadena de caracteres en formato IP, por ejemplo *192.168.1.10* (ver anexo del código) y por otra parte es necesario configurar la direccion IP de la computadora a la que se va a conectar la tarjeta, recordando que los tres primeros octetos deben de ser igual que como se configuró en el código y el úitimo puede ser cualquier valor direrente al configurado en el código, esto para que ambos dispositivos sean visibles dentro de la misma subred, finalmente la conexión queda como se ve en la figura 3.22

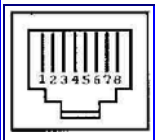
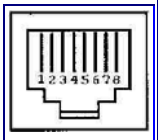
 Conector1	Color del cable	 Conector2
PIN 1	/NARANJA	PIN 3
PIN 2	NARANJA	PIN 6
PIN 3	/VERDE	PIN 1
PIN 4	AZUL	PIN 7
PIN 5	/AZUL	PIN 8
PIN 6	VERDE	PIN 2
PIN 7	/MARRÓN	PIN 4
PIN 8	MARRÓN	PIN 5

Figura 3.21 Configuración de cable cruzado

Al igual que la configuración con DHCP, en este esquema con IP fija una vez que la tarjeta es alimentada se muestra en el LCD la dirección IP con la que esta configurada, aquí no es necesario conectar el cable Ethernet ya que la IP queda configurada automaticamente.

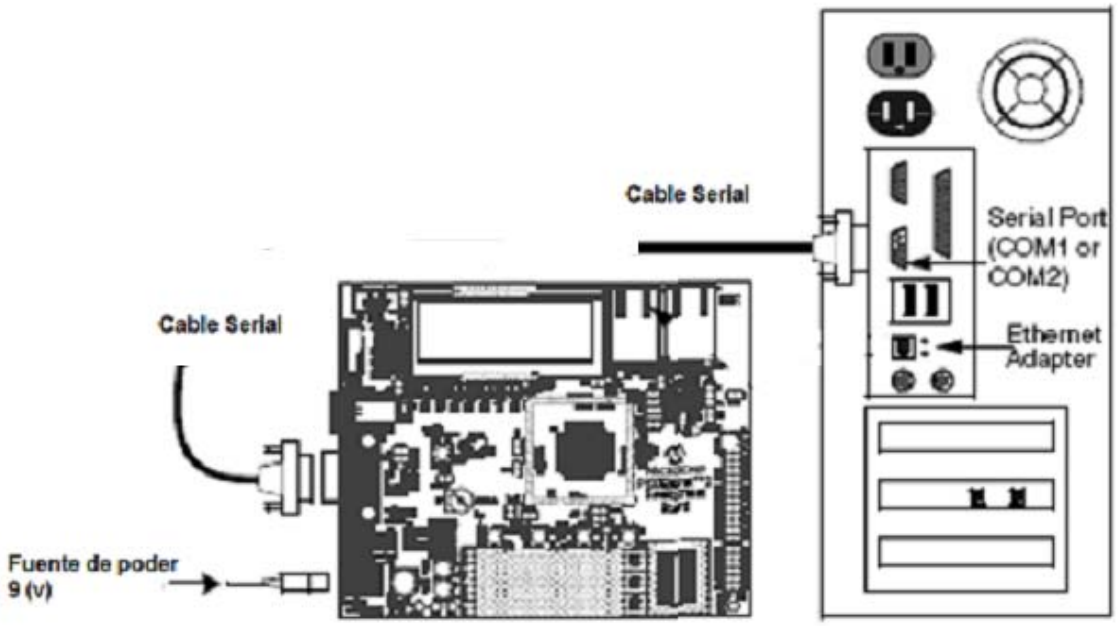


Figura 3.22 Conexión de la tarjeta con el cable cruzado

Una vez que se tiene la conexión se procede a ingresar a la página programada dentro de la tarjeta de desarrollo (MPFS.c), para ello basta con abrir el visualizador de internet y escribir la dirección IP con la que esta configurada la tarjeta de desarrollo, asi mismo se puede poner el nombre del archivo especifico al que se desea accesar o por default el archivo *index.html* si ese existe.

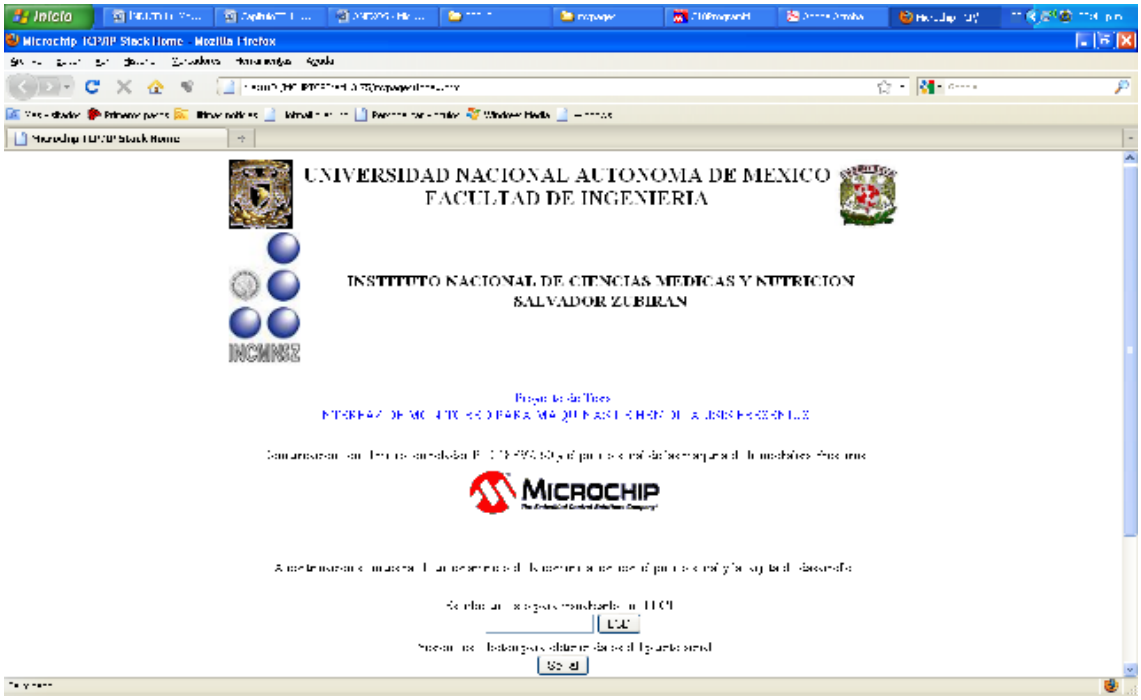


Figura 3.23 Página web programada en la tarjeta

3.8 Comunicación de los datos de la página Web con el microcontrolador

El siguiente paso es comunicar la página web con la tarjeta de desarrollo, en otras palabras es necesrio controlar las funciones de la tarjeta de desarrollo mediante la página web o con alguna invocación que provenga de está, por ejemplo se debe de hacer la lectura del puerto serial (máquina de hemodíalisis) únicamente cuando la aplicación web lo requiera, así mismo se deben de refrescar los valores solamente cuando la página asi lo requiera, en otras palabras está es la parte más importante del desarrollo en lo que corresponde a la parte de hardware, pues además de conjuntar los elementos descritos anteriormente tenemos que controlar todas las funciones de la tarjeta de desarrollo con la propia aplicación web.

Para lograr lo anterior tenemos que partir del hecho que tanto la página web como las funciones y variables que se utilizan en la tarjeta de desarrollo se encuentran almacenadas en la misma memoria de programa, es decir dentro de la memoria del microcontrolador, entonces es posible deducir que se pueden accesar y manejar los valores de manera relativamente sencilla, sin embargo es importante no

olvidar que lógicamente son cuestiones muy diferentes pues por un lado tenemos una aplicación web y por otro lado tenemos una aplicación embebida en un microcontrolador, pero pese a las dificultades de programación esto es posible gracias al uso de la Microchip Stack TCP/IP.

En primer lugar dado que el microcontrolador ejecuta su aplicación en tiempo real es necesario que la página web también puede obtener sus valores en tiempo real y para logra esto se hace uso de la página web también puede obtener sus valores en tiempo real y para logra esto se hace uso de la tecnología AJAX, recordemos que mediante esta tecnología se pueden actualizar datos de una aplicación HTML de manera casi instantanea con datos provenientes del servidor donde se ejecuta la aplicación, en este caso la tarjeta de desarrollo.

3.9 Desarrollo de la aplicación en el PICDEM NET 2

En el código fuente se deben declarar algunas variables estáticas las cuales se almacenan en la memoria del programa y conociendo la dirección de memoria de dichas variables es posible acceder a ellas mediante la página web, este procedimiento y configuración se explican a continuación.

Para nuestro caso en la carpeta que utilzamos para generar el archivo MPFS.c es necesario incluir un archivo html que contenga la funcionalidad de AJAX, es decir que haga uso del objeto *XMLHTTP* y todas sus funciones para generar la página dinámica, también se requiere de un archivo que contenga código que se comunique directamente con el servidor en nuestro caso con la tarjeta de desarrollo, dicho archivo contiene la programación necesaria para este fin, esto se logra mediante el uso de cadenas representadas por un valor hexadecimal que no son otra cosa que las direcciones de memoria de variables estáticas declaradas en el microcontrolador. (Ver anexo archivo cgi)

Figura 3.24 Diagrama a bloques de la interfaz de AJAX

Con el mecanismo anterior la página web actualiza dinámicamente los datos mostrados en aquellos elementos html que se programaron con el objeto *XMLHTTP* con datos provenientes de las variables estaticas del código de la tarjeta de desarrollo.

En el código de la tarjeta de desarrollo se deben de declarar las variables estáticas de la siguiente manera:

#define nombre_variable (direccion_memoria)

Lo anterior indica que se define una variable *nombre_variable* la cual se encuentra direccionada en la localidad que se coloca entre parenetesis, es decir para accesar a su contenido unicamente se requiere conocer la dirección de memoria que corresponde a los valores hexadecimales en el archivo que comunica el archivo html con la tarjeta.

Por otro lado se requiere el uso de dos funciones: *HTTPGetVar* y *HTTPExecCmd*, su funcionamiento se explica a continuación:

HTTPGetVar. Esta función sirve para recolectar los valores de todas las variables estaticas declaradas en el código de la tarjeta y enviarlas a la aplicación web para esto lo único que se hace es copiar una cadena de caracteres en la dirección de memoria correspondiente a la viarable declarada y automaticamente este valor se verá reflejado en la página web. (Ver anexo de código)

HTTPExecCmd. Esta función permite ejecutar alguna operación una vez que se da la orden desde la página web, esta orden se puede enviar mediante un formulario html que transmite sus elementos mediante AJAX en el cual se deposita el valor proveniente de formulario html en una variable estática del código de la tarjeta y mediante el uso de valores condicionales la tarjeta ejecuta alguna operación en nuestro caso la lectura del puerto serial. (Ver anexo de código)

Finalmente en nuestra aplicación la página web alojada en la tarjeta contiene un formulario html el cual le enváa sus valores al cádigo del microcontrlador de manera dinámica mediante AJAX y con el uso de la función *HTTPExecCmd*, este hace la lectura del puerto serial, guarda los datos en las variables estáticas ya declaradas y las muestra en la página web casi instantaneamente utilizando la función *HTTPGetVar* y la tecnología AJAX, de esta forma es posible refresacar los parámetros fisiológicos provenientes de la máquina de hemodíalisis.

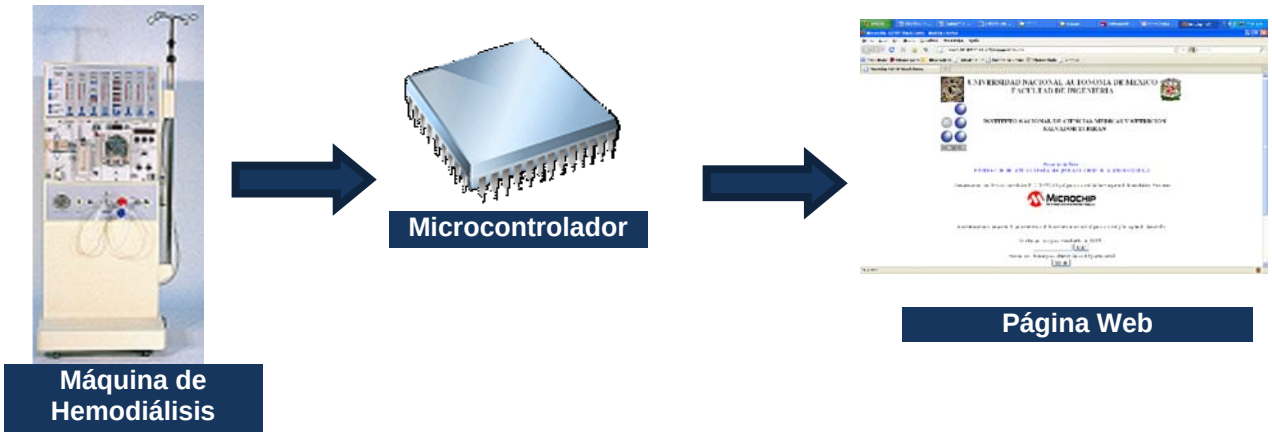


Figura 3.25 Diagrama a bloques la conexión final

3.10 Modelado de la Base de Datos

Para el modelo de la base de datos vamos a utilizar el modelo entidad-relación, el cual cumplirá con la Tercera Forma Normal.

El Diagrama Entidad Relación (DER) que representa nuestra base de datos.

a) Lógico (ver figura 3.26)
b) Físico (ver figura 3.27)

3.11 Modelado a capas

Se adoptará una arquitectura cliente servidor con un modelo a dos capas.
Dentro de nuestro sistema el PIC será nuestro servidor debido a que es el que contiene los parámetros obtenidos de las máquinas de hemodiálisis y la parte del cliente se encontrará con los clientes y las aplicaciones. Las capas a utilizar son la capa de datos y la capa de presentación.

La capa de presentación está representada por la interfaz y el uso de la tecnología AJAX en el momento de recepción continua de los datos (parámetros fisiológicos), por otra parte, la capa de negocios para este proyecto se representa con PHP.

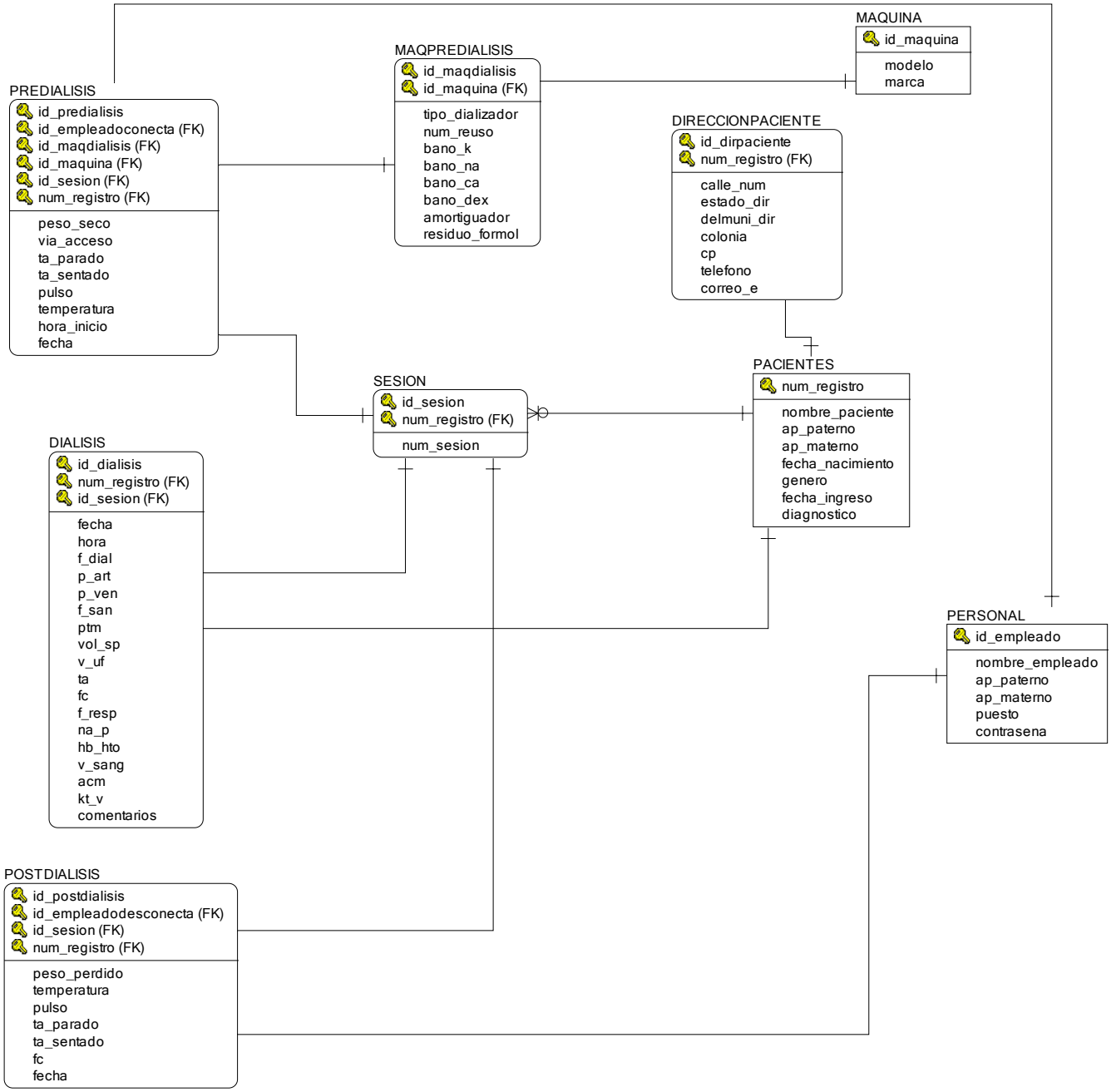


Figura 3.26 Diagrama Entidad Relación. Lógico

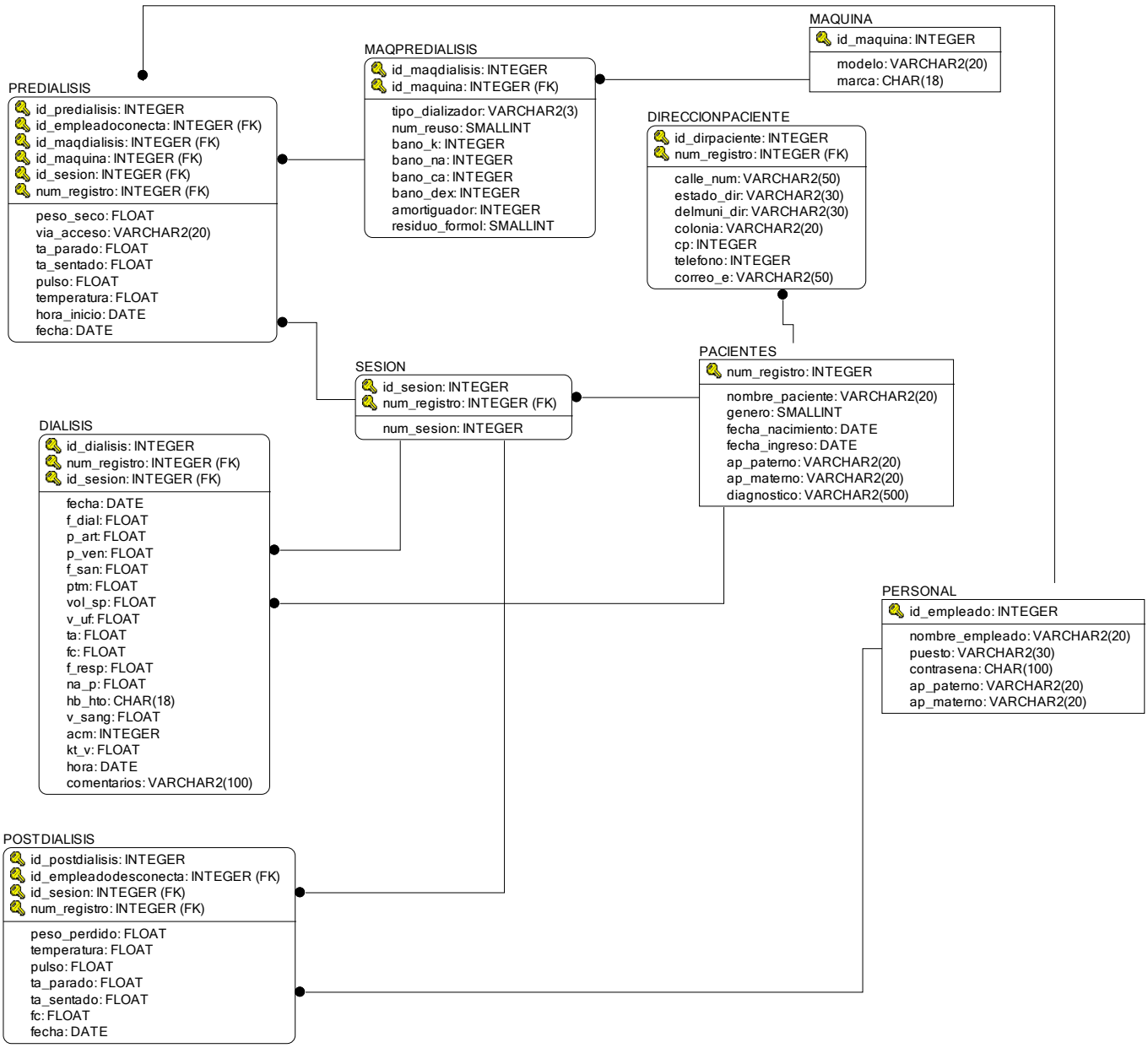


Figura 3.27 Diagrama Entidad Relación. Físico

Abreviatura	Significado
f_dial	Flujo de dializante
p_art	Presión en línea arterial
p_ven	Presión en línea venosa
f_san	Flujo sanguíneo
ptm	Presión transmembrana
vol_sp	Vol. de sangre procesada
v_uf	Vol. de ultrafiltrado
ta	Presión arterial
fc	Frecuencia cardiaca
f_resp	Frecuencia respiratoria
na_p	Sodio plasmático
Hb_hto	Hemoglobina/hematocrito
v_sang	Volumen sanguíneo del paciente
acm	Aclaramiento del dializador
kt_v	

Tabla 3.2 Significado abreviaturas de la base de datos.

3.12 Aplicación Web

Uno de los principales motivos por los cuales se ha decidido elaborar el centro de monitoreo a través de una aplicación web ese debe a la petición del Instituto Nacional de Ciencias Médicas y Nutrición.

El interés del Instituto en que el proyecto se desarrollará de este modo surge a partir de la actual necesidad del personal hospitalario. El personal necesita encontrarse obligatoriamente, durante cada sesión de diálisis, dentro del área metabólica para tomar las mediciones y verificar el estado del paciente. Implementando una aplicación web, las personas encargadas de anotar y analizar la información (parámetros fisiológicos) pueden consultar estos datos desde cualquier computadora que tenga acceso a la red.

Otra petición realizada por parte del INNSZ es que la aplicación utilizara software libre. Lo anterior con motivo de poder utilizar está aplicación sin realizar grandes gastos económicos.
Acatando los deseos de nuestro usuario final y después de una evaluación de características se decidió utilizar para nuestra aplicación, las siguientes tecnologías:

- PHP

- MySQL.
- AJAX

Un motivo por el cual se ha decidido utilizar estas tecnologías se debe a su popularidad a nivel mundial, sobre todo del grupo de tecnologías Apache, MySQL y PHP trabajando juntos dentro de aplicaciones web. Lo anterior se corroborará con estadísticas realizadas por dos empresas dedicadas al monitoreo y seguridad web (Pingdom y Netcraft), en estas estadísticas se reporta que el servidor web predilecto es Apache, por otro lado en el rubro de base de datos domina MySQL y en programación web PHP tiene la ventaja.**[61][62]**

Características importantes para la utilización de PHP es que es una tecnología de programación web open source, eficiente, multiplataforma, portable, soporta varios gestores de bases de datos además de ofrecer buena documentación.

Motivos por el cual se utiliza como gestor de bases de datos MySQL son: realiza una buena mancuerna con PHP, escalable, veloz, portable, open source.

Aunado a las tecnologías PHP, MySQL y Apache, los cuales hacen una excelente mancuerna trabajando los tres juntos se integra al desarrollo del centro de monitoreo una tecnología de diseño web, AJAX.

Los motivos de la utilización de la tecnología AJAX son:

- Para el desarrollo del centro de monitoreo, se necesita realizar peticiones continuas debido a los datos cambiantes; sin embargo, también se requiere que no se cargue la página completa, reduciendo así el tiempo de espera de petición.
- Basado en estándares abiertos. (JavaScript, HTML, XML, CSS).
- Funciona con los navegadores más populares y es compatible con cualquier tipo de servidor y lenguaje de programación Web (PHP, ASP, Perl, JSP, etc).

De las características más importantes que poseen en conjunto estas tecnologías y motivo por el cual se ha decidido su utilización son: open source, compatibles entre sí y tienen desempeño eficiente.

Otro aspecto importante para la toma de decisión fue que los hosting que gratuitos que existen en la red, manejan Apache, PHP y MySQL.

3.13 Interfaz gráfica

La interfaz de usuario final está compuesta principalmente por las siguientes ventanas:

a) Inicio de sesión (index.html)
b) Menú Principal
c) Registro Paciente
d) Registro Personal
e) Búsqueda Paciente
f) Búsqueda datos personales
g) Búsqueda de Historial Médico
h) Prediálisis
i) Diálisis
j) Postdiálisis

} Sesión

A continuación se explica brevemente el funcionamiento de cada uno de ellos.

a) Inicio de sesión

En este apartado el usuario (personal del hospital) se conectará, esto es una medida de seguridad para que no ingrese cualquier persona del hospital al sistema.

Para tener el nivel de seguridad en la contraseña se utilizó MD5.

Dentro de esta página si el usuario y contraseña son correctos el usuario inicia una sesión y lo envía al Menú Principal del sistema.

Si el usuario no ha introducido ningún valor en los campos de texto entonces no deja avanzar al usuario y muestra que es obligatorio introducir los datos dentro del campo.

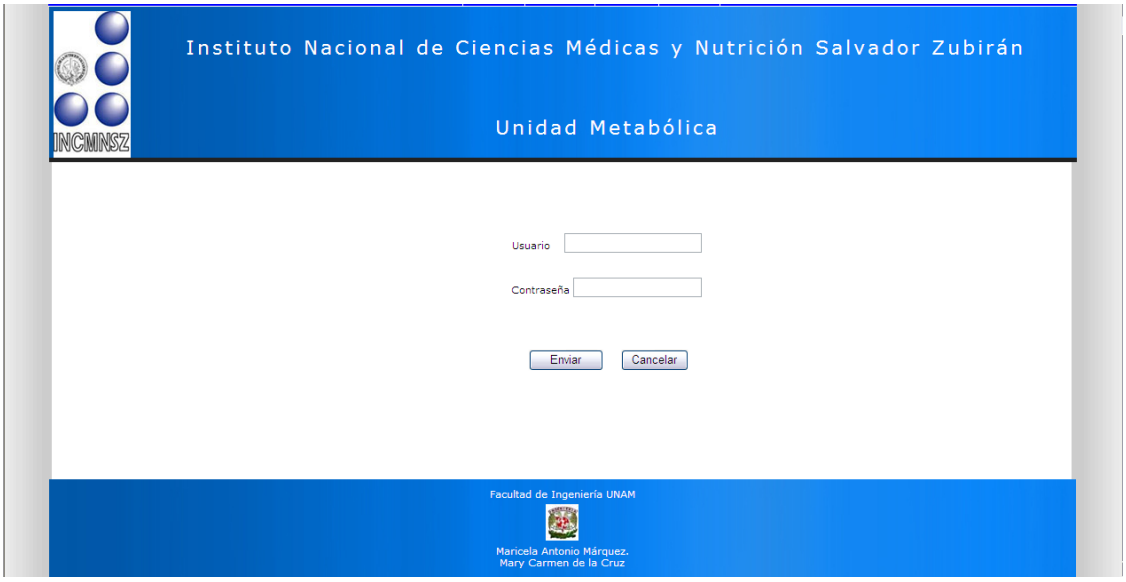


Figura 3.28 Ventana Inicio de sesión

En el menú principal del sistema se tiene acceso a diferentes partes del sistema de acuerdo a la acción que se desee realizar (Figura 3.29). Dentro de este se encuentran dos rubros

- Búsqueda

Dentro de búsqueda se encuentran las opciones para buscar a un paciente, datos personales del paciente y mostrar la información de sesión de un paciente.

En búsqueda paciente nos direcciona a la ventana de buscar un paciente previamente registrado (Figura. 3.32) y posteriormente iniciar una nueva sesión de hemodiálisis.

Dentro del link de búsqueda de datos personales se tendrá acceso a los datos de algún paciente en específico. (Figura 3.33)

Para búsqueda de historial médico se tiene acceso a los datos que ha manejado el paciente en sus diferentes sesiones, para ello también se realizará una búsqueda del paciente y la selección de diferentes opciones para mostrar los datos deseados. (Figura 3.34)

- Registro.

En el apartado de registro también se localizan dos opciones. La primera es registro paciente y la segunda registro de personal.

Al seleccionar registro paciente encontraremos una ventana donde se llenará un formulario con los datos principales del enfermo (Figura 3.30).

En registro de personal es el medio por el cual se registra a un nuevo empleado del hospital el cual podrá realizar sesiones de hemodiálisis. (Figura 3.31).

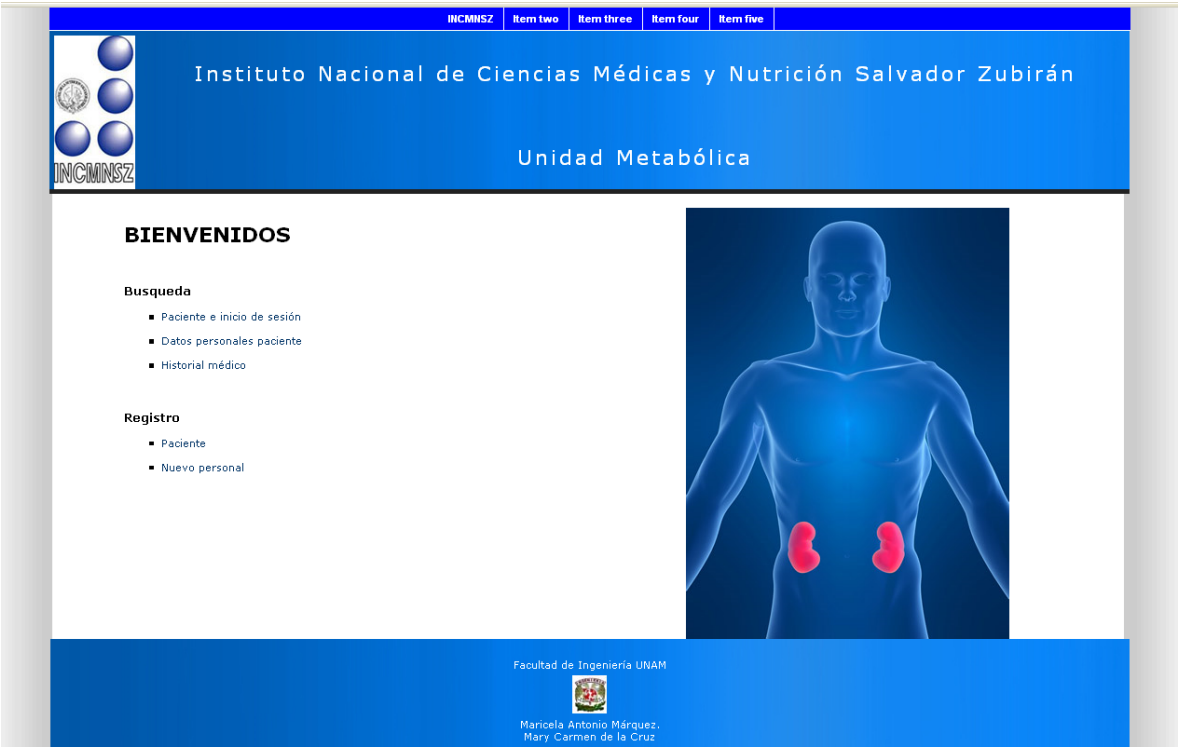


Figura 3.29 Ventana Menú Principal

b) Registro de pacientes

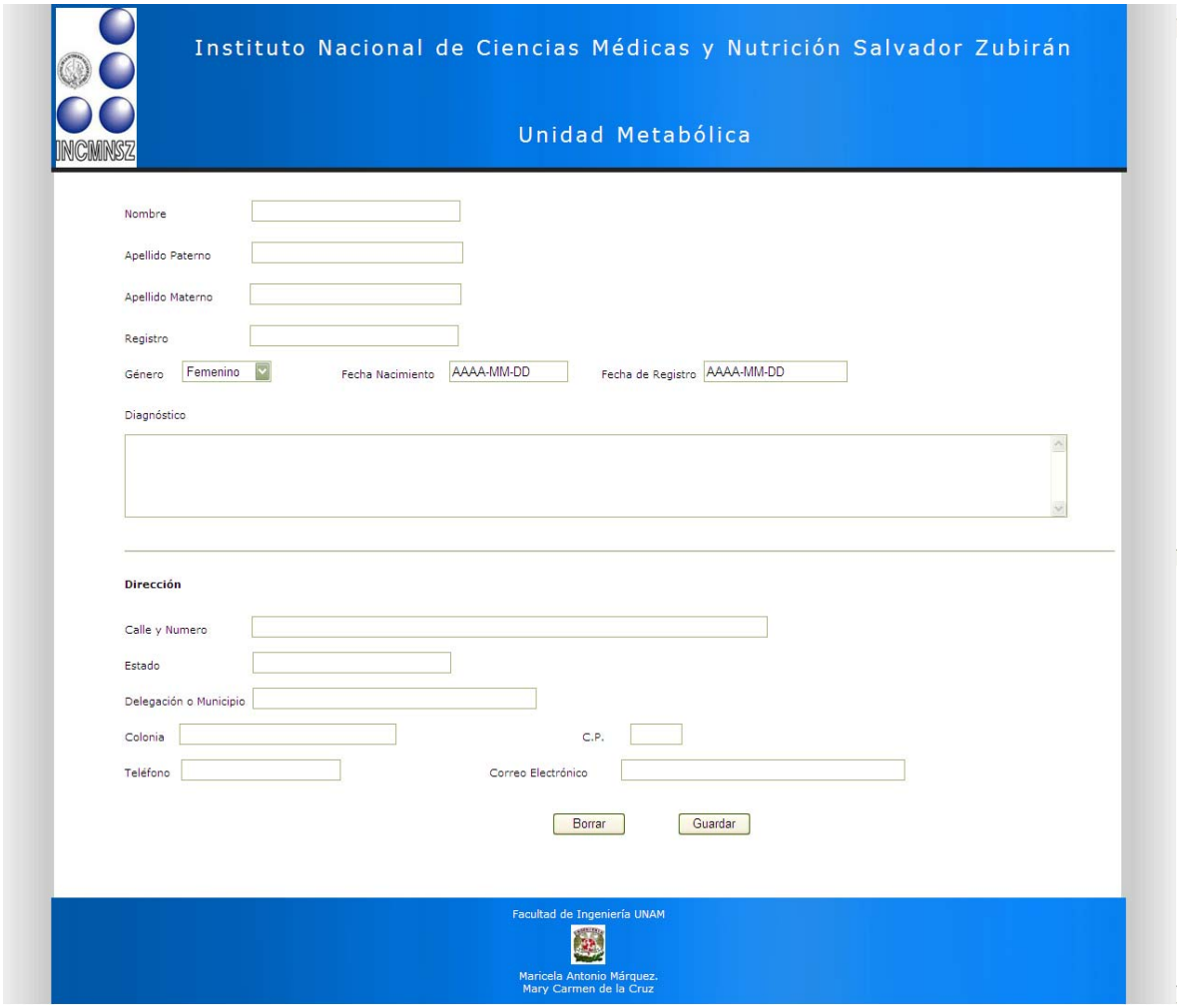


Figura 3.30 Ventana Menú de Registro de Pacientes

Dentro de la ventana de registro encontramos los datos principales y personales del paciente. El objetivo es que el paciente sólo se registre una vez y para las próximas sesiones de diálisis sólo se tenga que realizar acceso a la base de datos.

c) Registro personal

Figura 3.31 Ventana Registro Personal

Dentro de la base de datos existe una tabla donde se manejan los datos del personal que está involucrado en las sesiones de hemodiálisis. Esta tabla existe con el objetivo de tener una relación del personal que conecta o desconecta a determinado paciente, además de que sólo estas personas tienen acceso al sistema para observar el seguimiento de determinado(s) paciente(s)

d) Menú principal

e) Registro personal

Dentro de la base de datos existe una tabla donde se manejan los datos del personal que está involucrado en las sesiones de hemodiálisis. Esta tabla existe con el objetivo de tener una relación del personal que conecta o desconecta a determinado paciente, además de que sólo estas personas tienen acceso al sistema para observar el seguimiento de determinado(s) paciente(s).

f) Búsqueda Paciente

En la ventana de búsqueda paciente (Figura 3.32) se localiza, por medio del número de registro del paciente, a la persona que posteriormente será sometida a la sesión de diálisis.

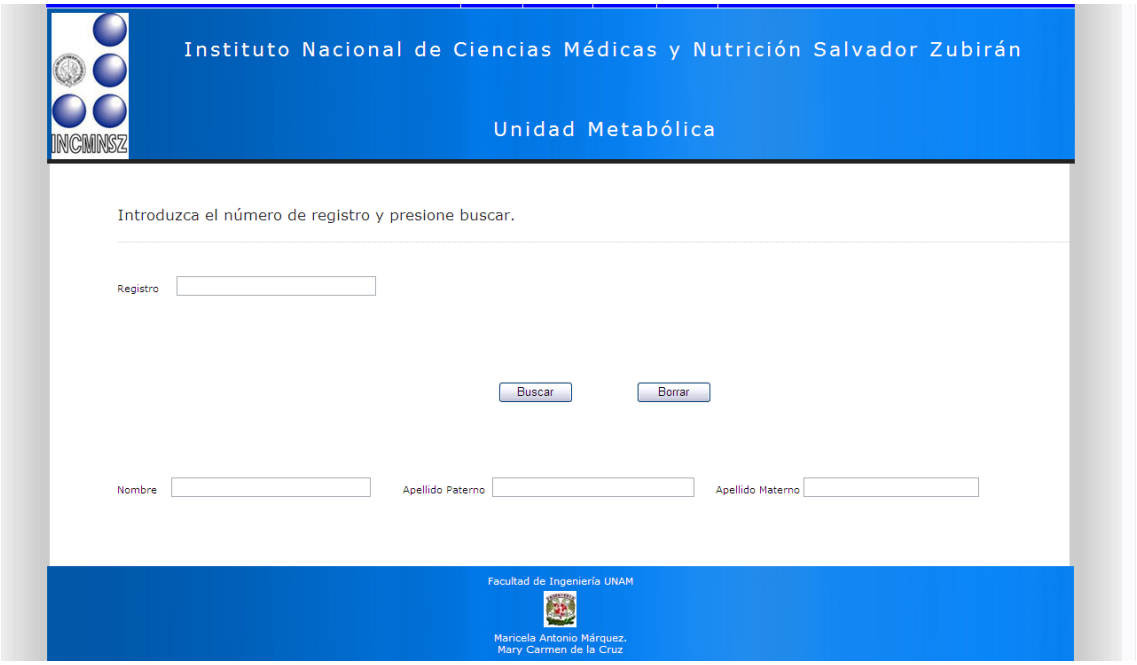


Figura 3.32 Ventana Búsqueda Paciente

g) Búsqueda de datos personales

En este apartado, al igual que en las ventanas anteriores, se localizarán los datos del paciente a través de su número de registro. Una vez que el usuario introduzca los datos se mostrarán en pantalla la información encontrada, mostrando así los datos personales del enfermo (nombre completo, dirección, teléfono, etc.)

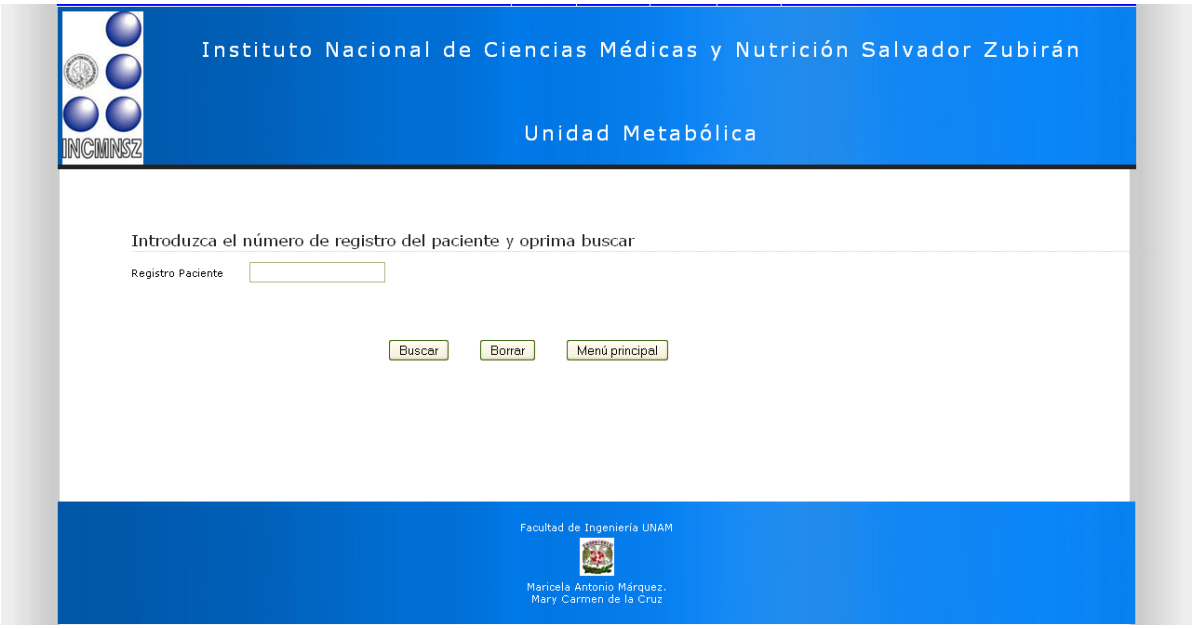


Figura 3.33 Ventana Datos Personales Paciente

h) Búsqueda Historial médico

Por medio de esta opción de Historial médico (Figura 3.34) el personal tendrá acceso a todos los parámetros fisiológicos guardados dentro de la base de datos de determinado paciente.

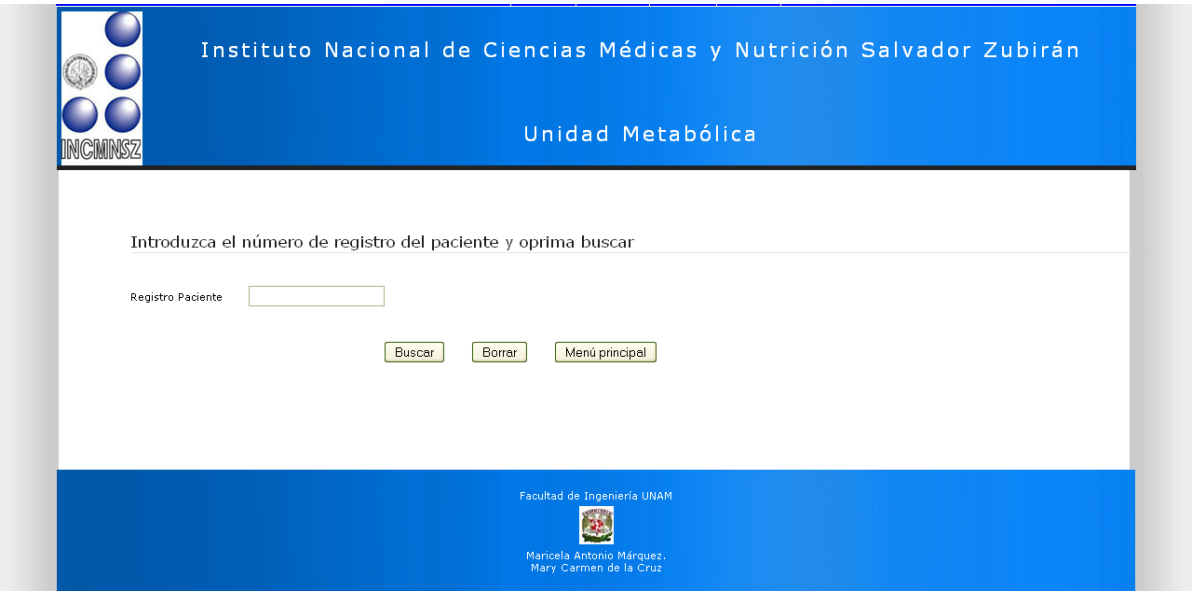


Figura 3.34 Ventana Historial Médico

Para la búsqueda del historial médico de determinado paciente se deberá introducir el número de registro del mismo, una vez presionando el botón buscar, se mostrará en pantalla los datos que se contengan del paciente.

i) Prediálisis.

Antes de la realización de la hemodiálisis a un paciente se necesita conocer algunos antecedentes del paciente como de la máquina a utilizar. Lo cual ayudará a conocer el estado previo y saber si es se puede realizar la hemodiálisis o no.

Dentro de la ventana de prediálisis se complementan estos datos de la máquina, como: prueba residuo de formol, baño K, Ca, Na, dex, etcétera. Por parte del paciente se necesita conocer parámetros fisiológicos para ser comparados al final de su sesión.

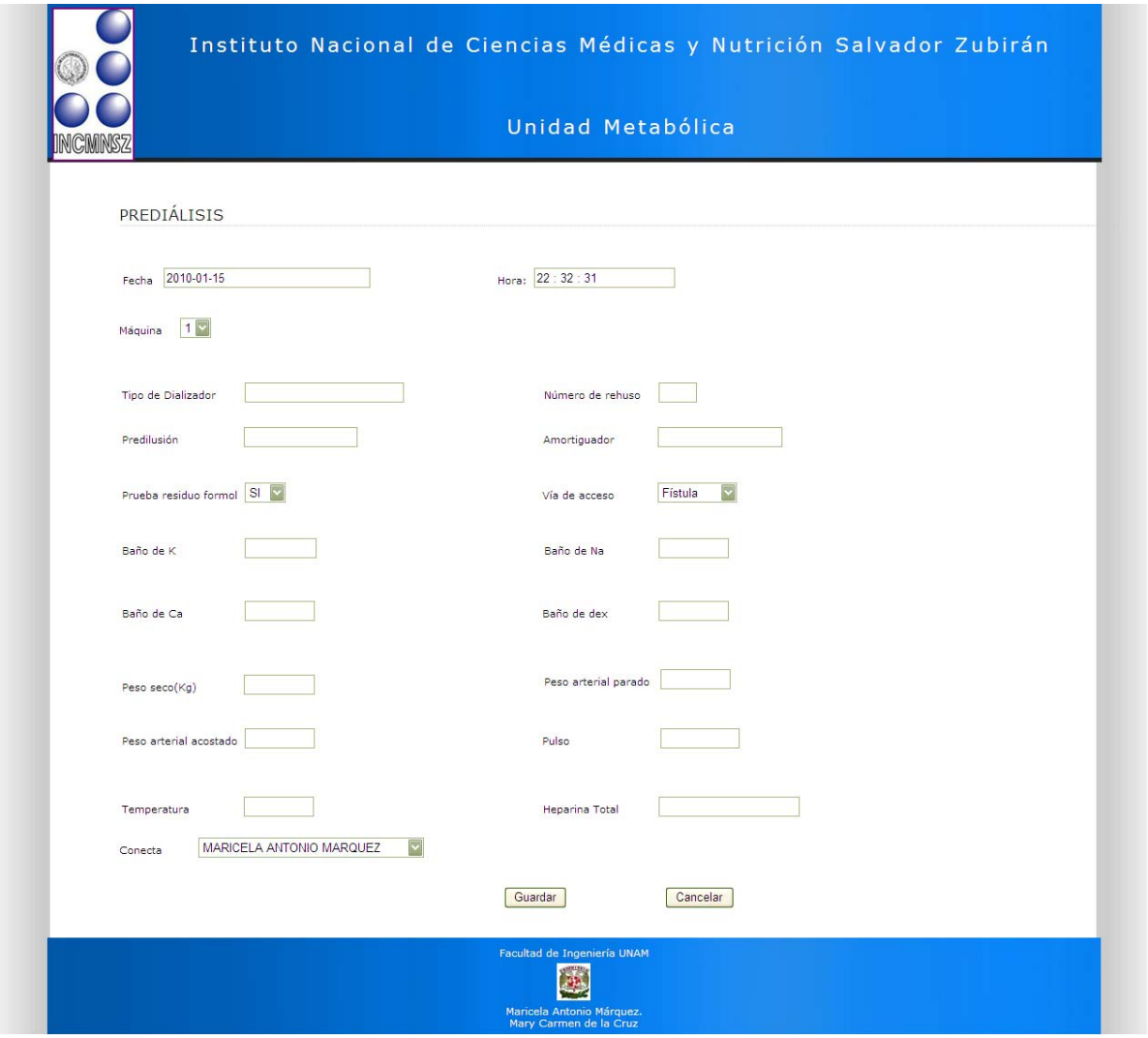


Figura 3.35 Ventana Prediálisis

j) Diálisis

La imagen 3.26 es una de las más importantes dentro del desarrollo del proyecto.

Dentro de esta ventana se observa durante toda la sesión de diálisis los parámetros fisiológicos que se van leyendo de la tarjeta de desarrollo (PICDEM NET2), la cual obtiene estos datos de la máquina de hemodiálisis.

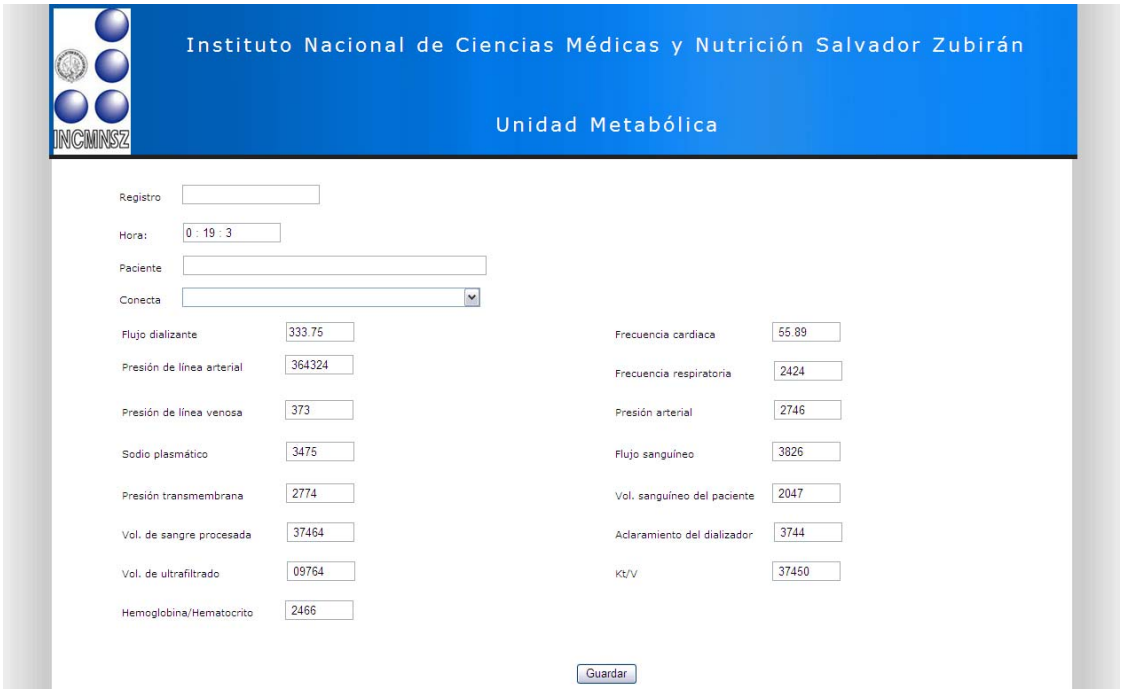


Figura 3.36 Ventana Dialisis

k) Postdiálisis

Figura 3.37 Ventana Postdiálisis

En la ventana de postdiálisis el personal coloca los parámetros fisiológicos finales del paciente. Esto para posteriormente, al consultar los datos, se realice una comparativa de su estado antes y después de la hemodiálisis.

Validación de campos.

Dentro del sistema se utiliza lenguaje javascript para realizar la validación de los campos. Esta validación es principalmente para que no se inserten valores nulos además de verificar el formato de los valores introducidos, ya sean valores numéricos, caracteres, etcétera.

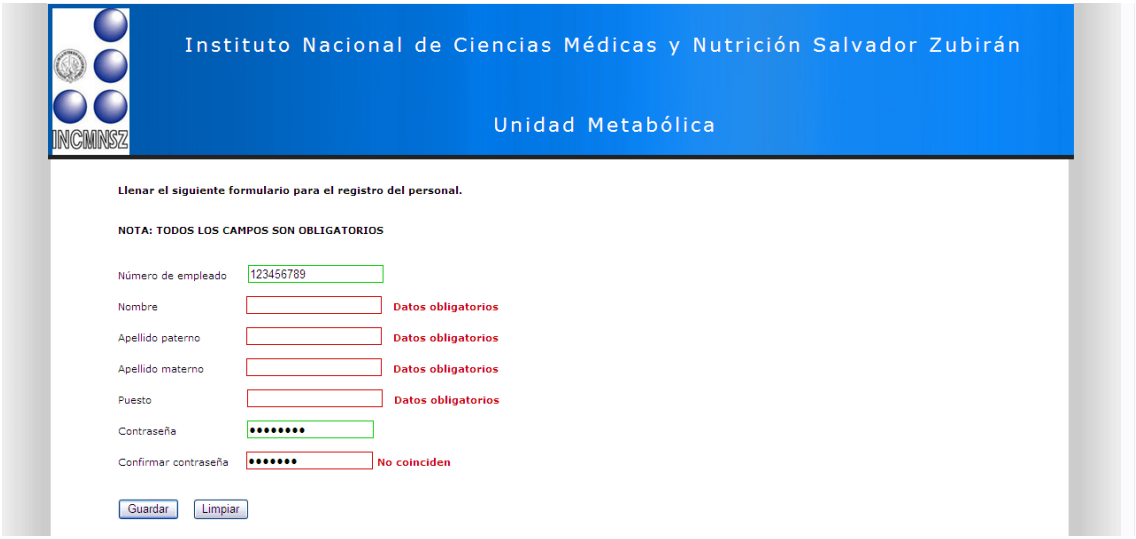


Figura 3.38 Ventana Validaciones LiveValidation

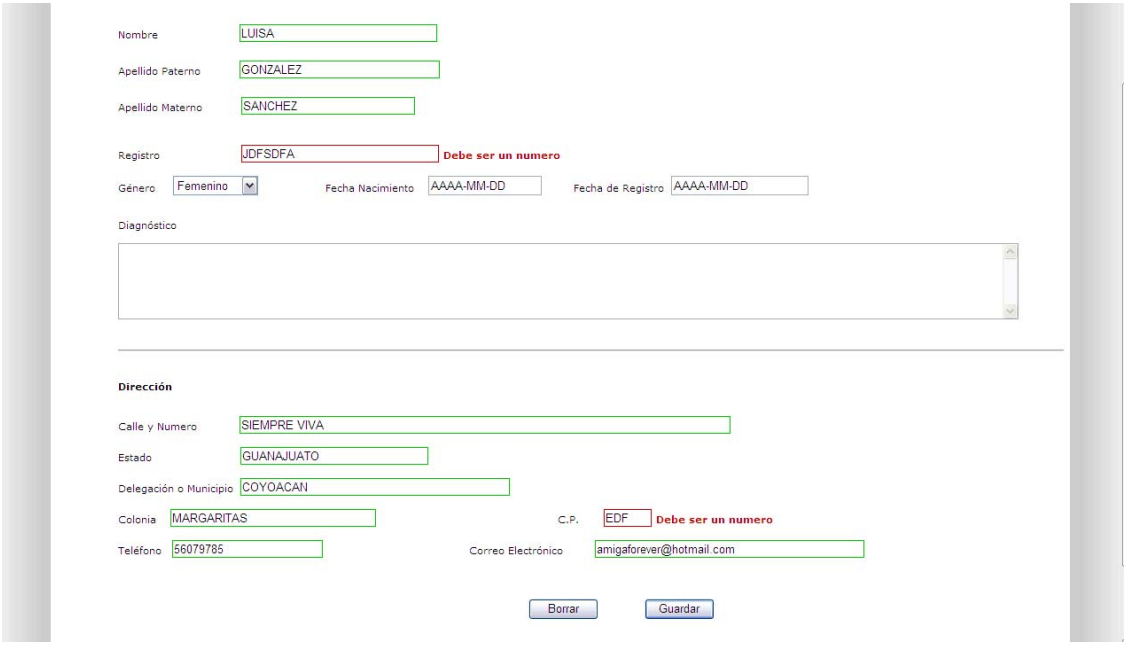


Figura 3.39 Ventana Validaciones LiveValidation 2

CAPÍTULO IV

PRUEBAS Y RESULTADOS

CAPÍTULO IV PRUEBAS Y RESULTADOS

4.1 Pruebas de Programación de la Tarjeta de Desarrollo

Dentro del desarrollo se hizo la prueba del funcionamiento de la tarjeta PICDEM NET 2, ejecutando la aplicación de ejemplo que viene pre cargada en ella, sin embargo es necesario conocer su funcionamiento y programación para nuevas aplicaciones. Así mismo dentro del desarrollo se comenzó con alguno programas en C18 con funcionamientos básicos de los elementos de dicha tarjeta, tal como son el manejo de puertos, el manejo del LCD, manejo y configuración del puerto serial y del puerto Ethernet, sin embargo dado que la base del programa está en la Microchip Stack TCP/IP fue necesario conocer perfectamente su funcionamiento y por tanto hacer una recopilación del código usado con el fin de realizar las modificaciones necesarias para nuestro caso.

El primer problema al que nos enfrentamos fue que al hacer la compilación del código fuente de la Microchip Stack TCP/IP, es decir del proyecto *C18ProgramMem.mcp* en MPLAB y con el compilador de C18 ya instalado correctamente fue el siguiente error:

> D:\MCHPTCPStack 3.75\Include\Compiler.h:1003:Error [1099] "Hardware profile not defined. See available profiles in Compiler.h. Add the appropriate macro definition to your project [ex: Project -> Build Options... -> Project -> MPLAB C18 -> Add (macro) -> HPC_EXPLORER]"

Al investigar en el código se logro averiguar que se debía de incluir la declaración de una macro dentro de las opciones de compilación del proyecto, con el fin de identificar que el programa se va a ejecutar en el PICDEM NET 2, lo anterior se debe a que la Microchip Stack TCP/IP, esta diseñada para poder utilizarse en diferentes tarjetas de desarrollo y familias de microcontroladores gracias al uso de directivas de preprocesamiento declaradas dentro del código. Para solucionar este problema basta con declarar la macro en la opción del menú Projec→Buil Options→Project, en la cual se abre una ventana como se muestra a continuación.

Seleccionando la pestaña MPLAB C18, en esta parte se da clic al botón Add y se escribe el nombre de la macro que hace mención a la tarjeta de desarrollo la cual queda como sigue: *PICDEMNET2.*

También es importante declarar una macro adicional, dado que nuestra propuesta de desarrollo no se va a utilizar memoria externa y que todo lo vamos a ejecutar desde la memoria de programa en el microcontrolador PIC18F97J60, entonces esto también se debe indicar en las opciones de compilación del proyecto, y para ello se declara la macro *MPFS_USE_PGRM,* de la misma manera que para la macro

anterior, lo que se le indica al compilador es que la imagen MPFS se encuentra alojada dentro de la memoria de programa.

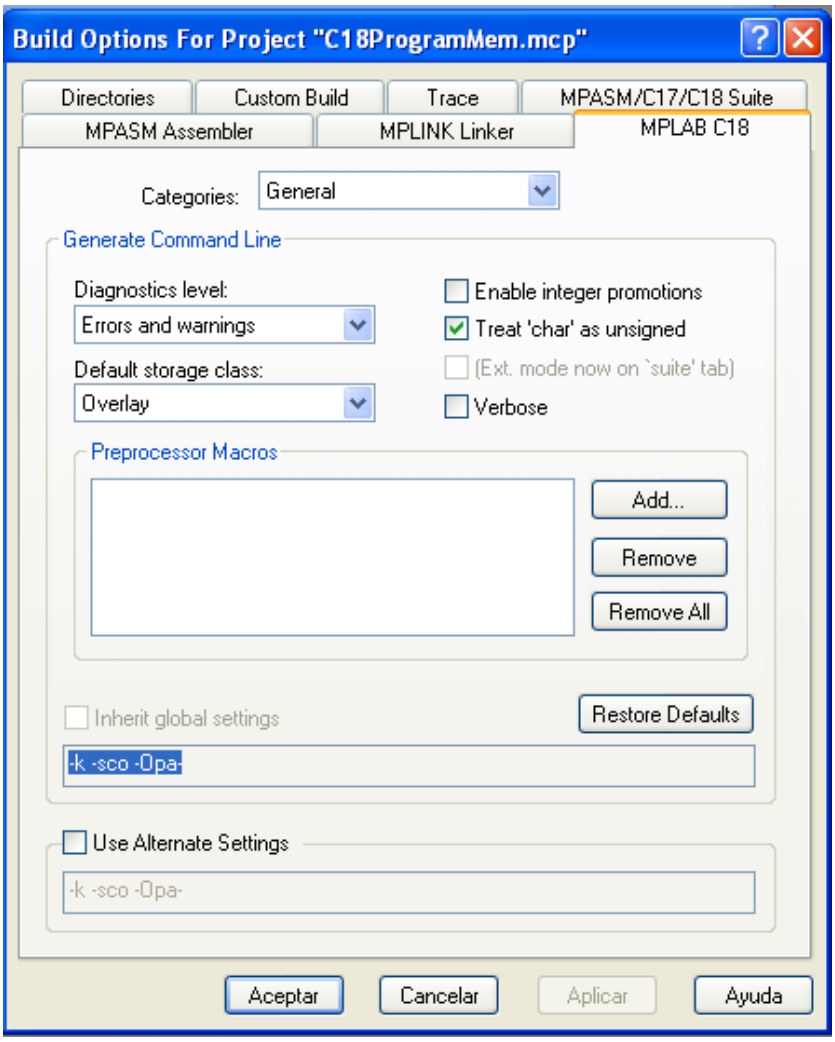


Figura 4.1 Configuración de la Macro en MPLAB C18

Otra parte importante en el desarrollo es el funcionamiento de LCD de la tarjeta, aunque este elemento no forma parte de la propuesta de desarrollo, ni de la propuesta de producto final como resultado del presente proyecto, durante el desarrollo y pruebas del prototipo que se está manejando representa un parte muy importante como interfaz de señalización es por esto que es importante conocer y aplicar su funcionamiento correctamente, para ello se tomo como base la librería *xlcd.h* que viene incluida dentro del compilador de Microchip C18, sin embargo existen algunos problemas al utilizar esta librería y las funciones de inicialización no trabajaron correctamente al momento de hacer nuestras pruebas, por ello se recurrió a la programación de nuestras propias librerías de uso del LCD, basándonos en lo siguiente:

Basándonos en el diagrama de la Figura 4.2, el puerto de datos del LCD se encuentra conectado físicamente al puerto E del microcontrolador PIC18F97J60 y los pines de control ENABLE, R/W y

RS, están conectados al bit 0, 1 y 2 del puerto H respectivamente, así mismo el dispositivo LCD que se encuentra en la tarjeta es compatible con cualquier modelo genérico del LCD, así como con sus comandos de configuración.

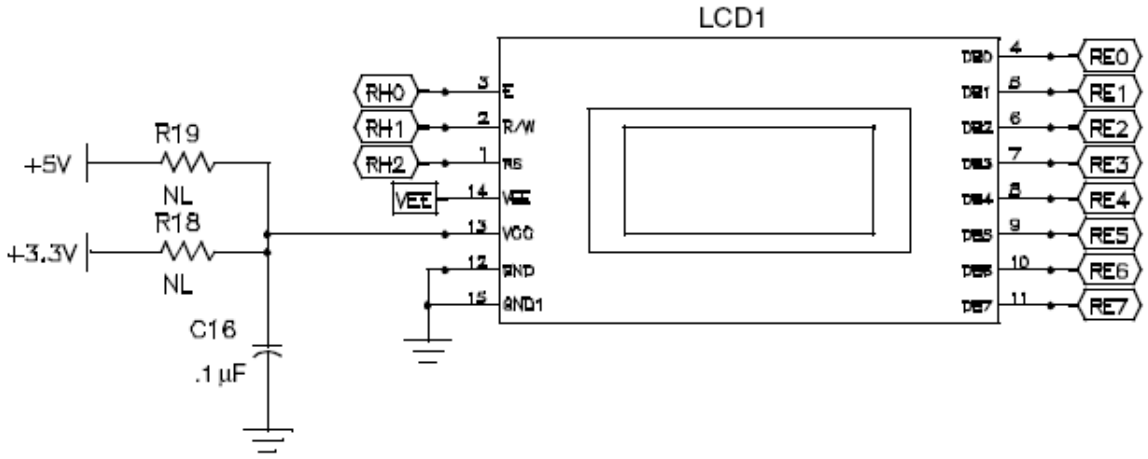


Figura 4.2 Conexión de los puertos del LCD

Dadas las condiciones de hardware de la tarjeta de desarrollo y el tipo de dispositivo LCD que se tiene (16X2), el diseño y programación de nuestra librería de manejo del LCD queda de la siguiente manera: Modo de configuración de 8 bits, tamaño de los caracteres 7X5, desplazamiento del cursor de izquierda a derecha, cursor activo y visible, las direcciones para la primera línea quedan configuradas desde la dirección 40H hasta la 4FH y para la segunda línea desde la 00H hasta la 0FH **[63]**, la secuencia de comando para lograr esta configuración es la siguiente:

Comandos de inicialización

- 38H
- 0EH
- 06H

Nota el bit R/W debe de estar en 0, ENABLE también debe de estar en 0 y RS debe de estar en 0, el tiempo para cada comando es aproximadamente de 1.5 a 2 (ms).

Comandos adicionales

- 01H (borrar pantalla)

Escritura de caracteres

- Para la primera línea 80H hasta 8FH
- Para la segunda línea desde C0H hasta CFH

Nota: Los bits RW y ENABLE deben estar en 0 y RS en 1, el tiempo para escribir un carácter es aproximadamente 1.5 (ms)

Teniendo lo anterior como referencia solo se programan las funciones para inicializar el LCD y escribir los caracteres (ver apéndice del código).

4.2 Pruebas de la Comunicación Serial

Para el manejo de la comunicación Serial de la tarjeta de desarrollo, la programación tiene como base la librería *USART.h* de Microchip C18, debido a que estamos utilizando una comunicación asíncrona en el puerto 1 de la comunicación serial (EUSART1), sin embargo al igual que con la librería del LCD las funciones que vienen por default no funcionaron correctamente por lo que se tuvo la necesidad de diseñar y programar las propias, para ello se utilizaron las configuraciones de acuerdo a la hoja de especificaciones del microcontrolador PIC18F97J60 las cuales son como sigue:

En la sección correspondiente a la configuración de la comunicación serial denominada EUSART, la cual se puede encontrar como *ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)* se encuentran las configuraciones para el manejo del puerto serial del microcontrolador, es importante destacar que este microcontrolador cuenta con dos interfaces de comunicación serial EUSART1 y EUSART2, de acuerdo al diagrama de la tarjeta de desarrollo PICDEM NET 2 que se muestra en la **Figura 4.3** la comunicación serial se realiza por el puerto EUSART1 que se encuentra en los pines 6 y 7 del puerto C de microcontrolador PIC18F97J60, además para calcular el rango de generación de baudios (BRG) se debe utilizar la siguiente fórmula:

BRG = (FRECUENCIA_MICROCONTROLADOR / RANGO_BAUDIO) / 16) - 1

La frecuencia de trabajo a la que esta configurado el microcontrolador en la tarjeta de desarrollo PICDEM NET 2 es de 40 (MHz), y el rango de baudios por default en la Microchip StacK TCP/IP es de 19200 (bps).

Además es necesario configurar los registros TXSTA1, RCSTA1 y BAUDCON1 con el fin de que se utilice una configuración de 8 bits, comunicación asíncrona, sin bit de paridad ni bit de parada, para ello se configura con los siguientes valores: TXSTA1, debe valer 04H y BAUDCON1 debe valer 00H, lo anterior se programa dentro de la función de inicialización de la tarjeta de desarrollo (ver apéndice de código)

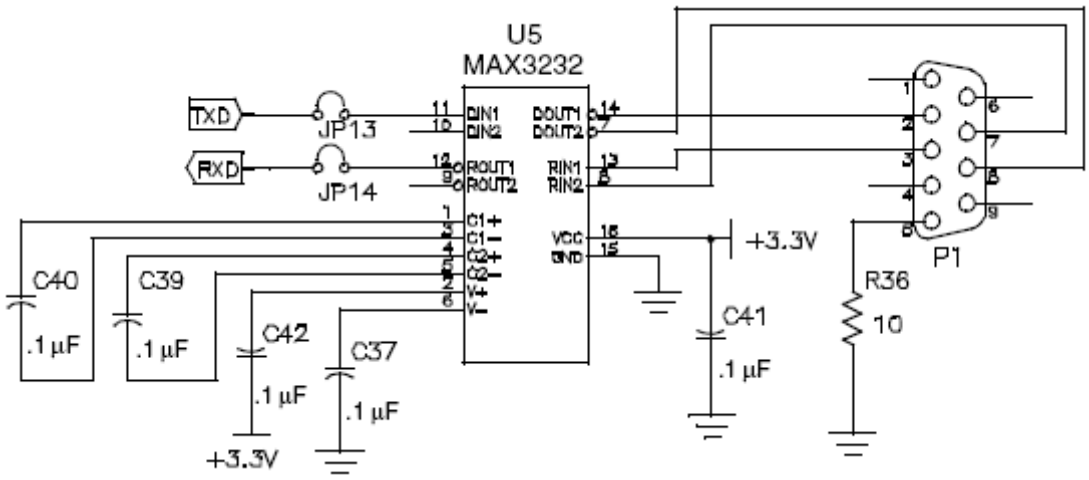


Figura 4.3 Comunicación serial

Dentro de nuestras pruebas se utilizaron todos los rangos de baudios disponibles en la aplicación Hyperterminal de Microsoft Windows que van desde los 110-921600 (bps) además se configuró los caracteres de entrada en ASCII en modo de 8 bits sin bit de paridad y en modo asíncrono tal como se muestra en la **figura 4.4**

Figura 4.4 Configuración en la Hyperterminal

Inicialmente no se mostraba ningún valor en el Hyperterminal sin embargo al cambiar la configuración en la entrada de caracteres con eco ya se pudieron apreciar los valores y los datos provenientes del puerto serial, adicionalmente se configuro el LCD de la tarjeta de desarrollo para que mostrará los caracteres provenientes del puerto serial tal como se muestra en la **figura 4.5**

En esta parte se tuvo el problema de que las cadenas de caracteres que se envían al puerto serial y al LCD deben de ser cadenas estáticas, sin embargo es necesario utilizar una variable que vaya cambiando sus valores, para corregir este problema únicamente se necesito utilizar la función *strcpy()* en lugar de la función *strcpypgm2ram()*, que es la función que se utiliza para copiar una cadena estática a una variable, por otro lado la función *strcpy()* copia el contenido de una variable a otra variable (Ver apéndice de código)

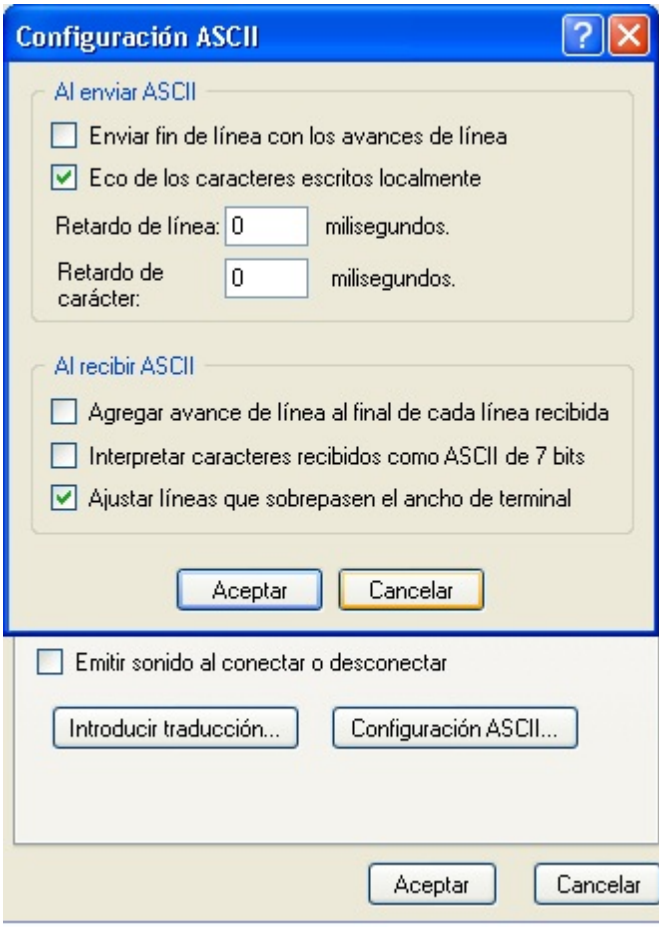


Figura 4.5 Captura de datos desde el Hyperterminal

4.3 Pruebas de interfaz Web en el microcontrolador

En el capítulo de desarrollo se explico cómo implementar la página web dentro de la tarjeta de desarrollo en base al uso de la imagen MPFS, las primeras pruebas que se realizaron fueron con páginas totalmente hechas en HTML estático y mostrando algunas imágenes y formularios, el resultado fue

satisfactorio, también se logro apreciar cómo se conforman estos elementos HTML dentro del archivo MPFS.c correspondiente a la imagen generada y que posteriormente es programada en el microcontrolador, las páginas HTML dentro de la imagen MPFS son codificadas en un arreglo de valores hexadecimales, generando un arreglo por cada archivo que se incluye en la imagen MPFS, es decir se crea un arreglo por cada archivo HTML, imagen, hoja de estilo, etc. Tal como se muestra en la figura 4.6

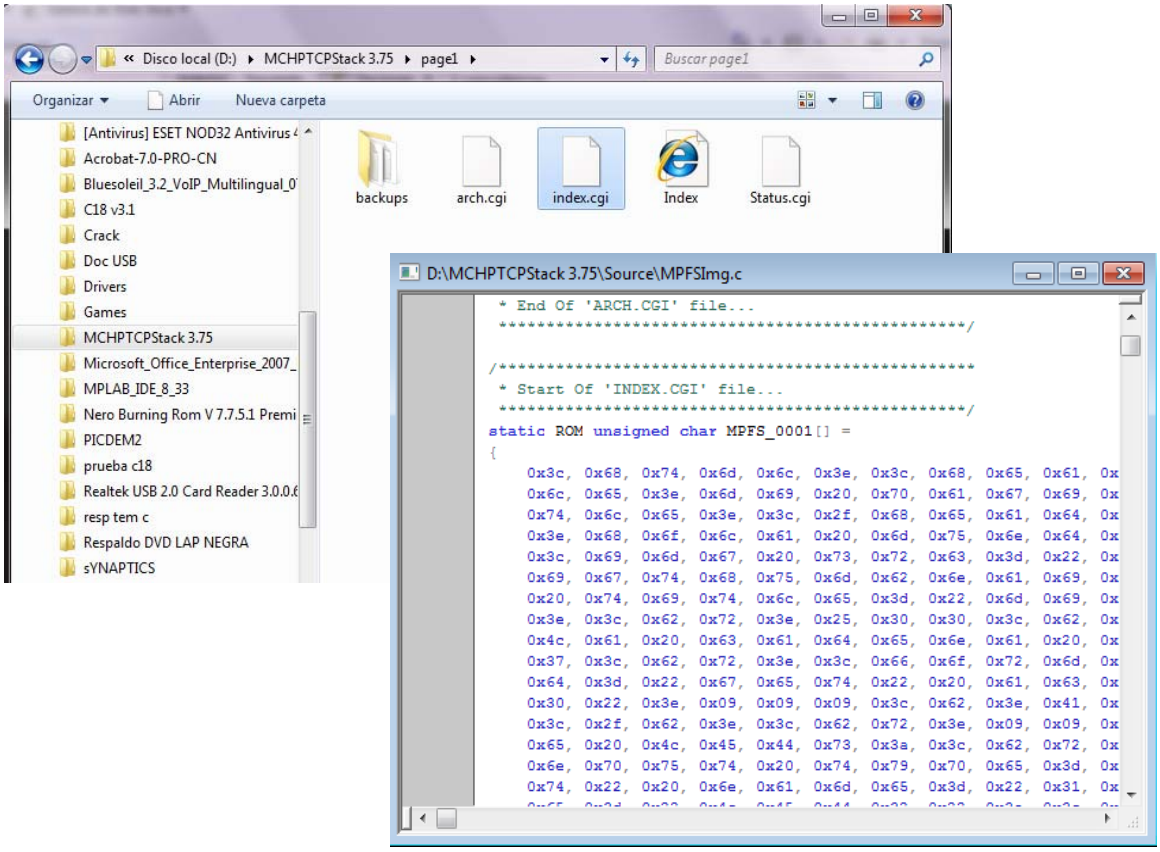


Figura 4.6 Imagen MPFS

Sin embargo los problemas comenzaron las páginas dinámicas, la Microchip Stack TCP/IP sugiere que los contenidos dinámicos que se comunican directamente con la memoria de la tarjeta de desarrollo sean archivos con extensión CGI, pero estos archivos no están codificados en el lenguaje nativo de CGI (PERL ó C) sino que es una codificación especial, donde los valores hexadecimales correspondiente a las direcciones de memoria de las variables que se desean utilizar son escritos en formato “%” seguido del numero hexadecimal tal como se muestra en la **figura 4.7.**

Entonces es necesario mezclar este código con el código HTML normal a fin de darle funcionalidad a las páginas web, es importante recordar que las páginas dinámicas pueden ser en cualquier tecnología web como por ejemplo PHP, JSP ó ASP, sin embargo para ejecutar estas tecnologías es necesario que el servidor sea capaz de interpretarlas, en nuestro caso el servidor es el

microcontrolador embebido en la tarjeta el cual carece de cualquiera de las tecnologías antes mencionadas y a cambio de ello proporciona una tecnología alternativa para el manejo de HTML dinámico.

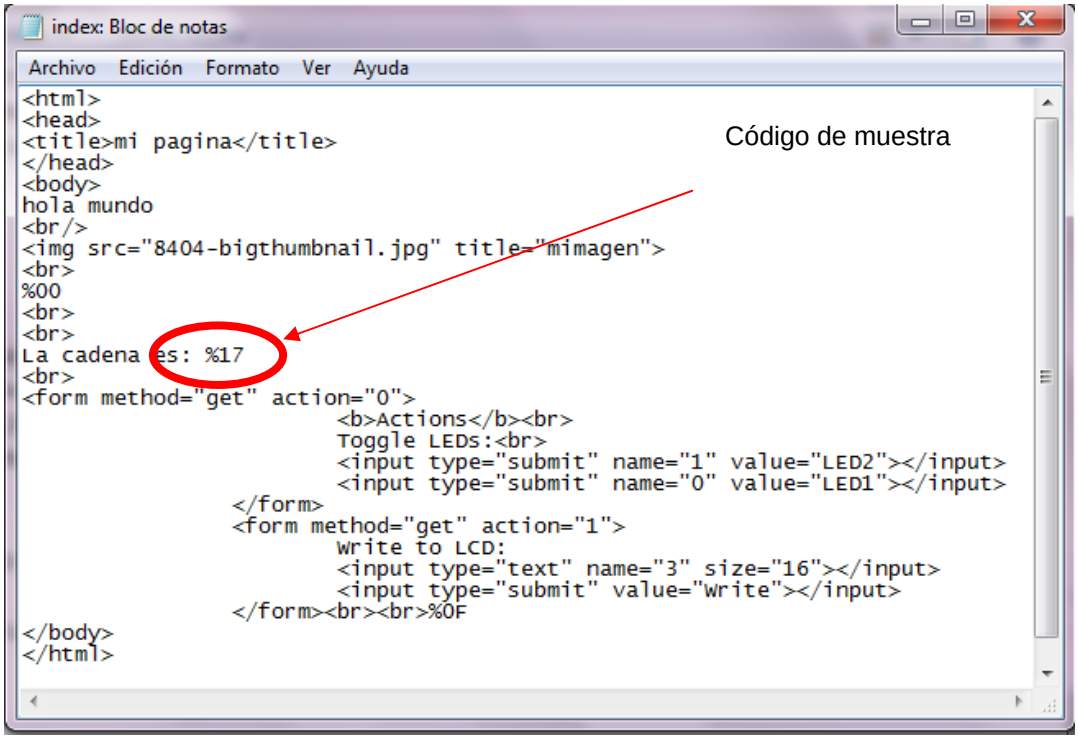


Figura 4.7 Muestra de formato de archivo dinámico

Es importante que los archivos que tienen la funcionalidad dinámica tengan extensión CGI, de lo contrario carecerán de funcionalidad, esto se logro apreciar en las pruebas realizadas con la tarjeta de desarrollo, pues si en la imagen MPFS se incluía algún archivo cuyo código era el indicado para una página dinámica pero la extensión es diferente de CGI esta página perdía totalmente sus funcionalidad tal como se muestra en la **figura 4.8.**

En base a lo anterior se tiene que los formatos manejados en el código de la Microchip Stack TCP/IP utilizados para nuestros propósitos son:

- HTML Para páginas estáticas con contenido AJAX
- GIF/JPG Para imágenes
- JS Para java script
- CSS Para hojas de estilo en cascada
- CGI Para contenidos dinámicos

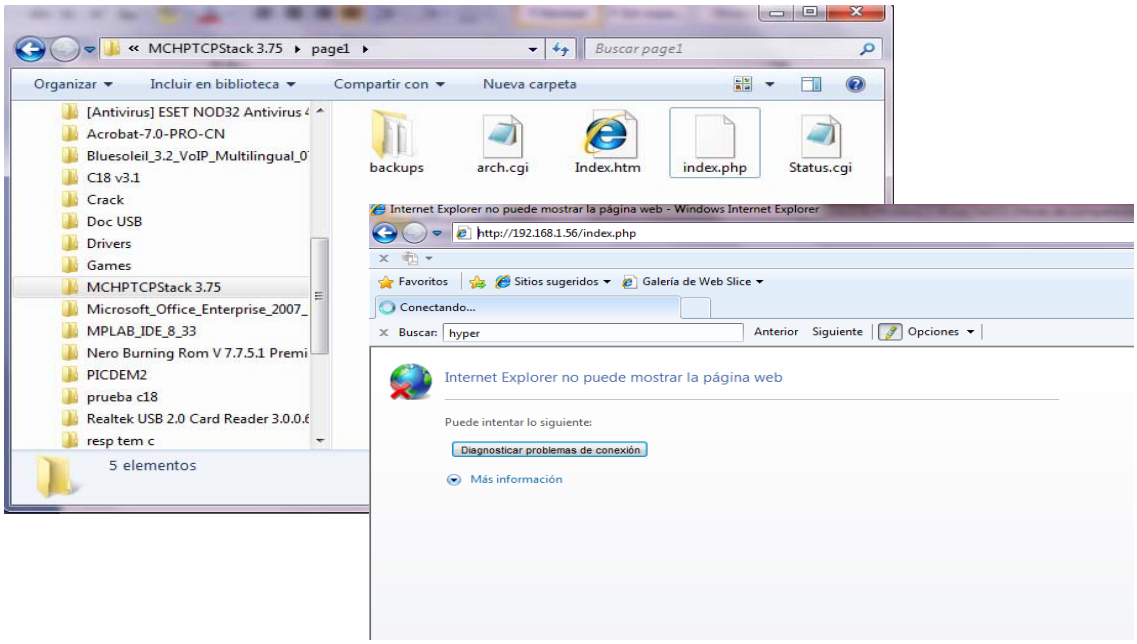


Figura 4.8 Error en el formato del archivo

Recordando también que todos los archivos que se vayan a utilizar en la imagen MPFS deben de estar en el mismo subdirectorio o de lo contrario no se tendrá la funcionalidad esperada.

Finalmente se hicieron las pruebas del manejo de la tarjeta de desarrollo desde la página web, inicialmente se tuvieron problemas al momento de enviar los datos desde el formulario HTIML hacia el código del microcontrolador, sin embargo esto se pudo solucionar fácilmente al identificar la sección de código donde se hace la comparación de los parámetros provenientes de la página web, esto dentro de la función *HTTPExecCmd ().*

La prueba final consistió en hacer un formulario HTML que contenía un campo de texto en el que se escribe una cadena de caracteres y al enviar los parámetros del formulario la cadena se refleja en el LCD de la tarjeta de desarrollo tal como se muestra en la **figura 4.9.**

4.4 Pruebas de interpretación de datos provenientes del serial hacia la interfaz Web

Esta sección es continuación inmediata de la sección anterior, pues si los datos ya se pueden reflejar en la página web y al mismo tiempo enviarlos de regreso a la tarjeta de desarrollo los siguiente es obtener datos de la interfaz del puerto serial y mostrarlos en la página web, para realizar estas pruebas se utilizó nuevamente el Hyperterminal de Microsoft, donde se escribe una cadena de caracteres y esta es enviada a la página web a través de la tarjeta de desarrollo.

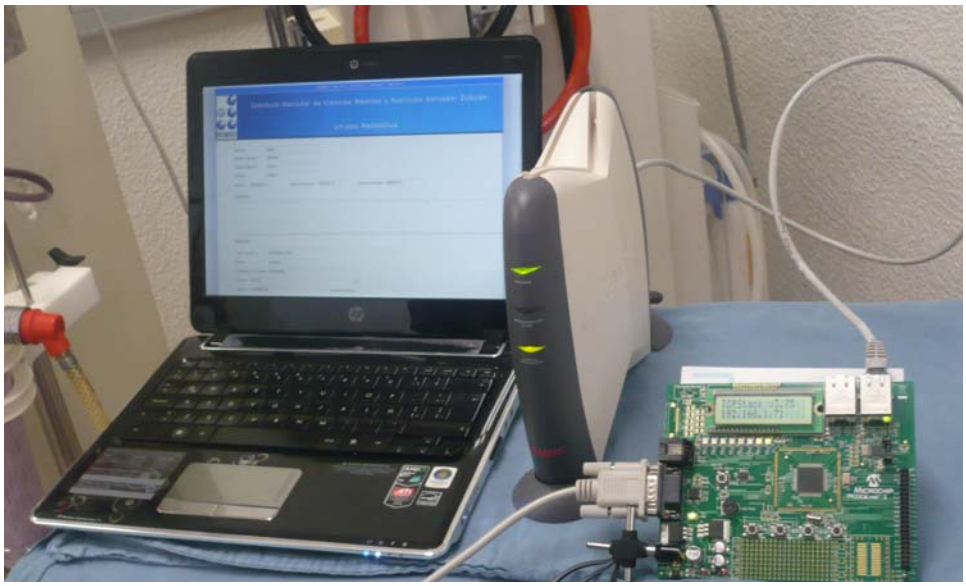


Figura 4.9 Envió de caracteres desde la página web hacia el LCD

Inicialmente se genero una aplicación en donde se tiene un ciclo que se encuentra leyendo constantemente el puerto serial y asociado a una variable, cada vez que se escribe algo en el hyperterminal la cadena se guarda en la variable anterior y automáticamente es mostrada en una página web, el problema que se tuvo en esta parte es que si el programa se cicla se pierde la funcionalidad de los demás elementos de la tarjeta y esta se queda bloqueada sino no hay ninguna cadena en el puerto serial, para resolver este problema se opto por implementar un Timer que sirviera como ciclo de espera para leer el puerto serial, es decir cada vez que el Timer se agota se dispara una interrupción que invoca a la lectura del puerto serial dentro del programa que se está ejecutando en la tarjeta de desarrollo, una vez hecho lo anterior el valor se transmite a la página web donde se actualiza inmediatamente gracias a la funcionalidad de AJAX tal como se muestra en la **figura 4.10**

El siguiente paso es invocar la lectura del puerto serial desde la página web, para ello se utiliza un botón en un formulario HTML que envía un parámetro a la tarjeta de desarrollo y cuando esta lo recibe envía una cadena al puerto serial que se refleja en el Hyperterminal y donde se espera que se escriba una cadena la cual es leída por la interfaz serial de la tarjeta y enviada automáticamente a la página web, en otras palabras se sustituye el uso del Timer con una invocación proveniente de la página web, pues esta la que debe de activar la funcionalidad de la lectura del puerto serial, que dando independiente el control de la tarjeta de desarrollo, lo anterior se logro correctamente tal como se muestra en la siguiente **figura 4.11.**

Figura 4.10 Obtención de datos de forma automática desde el puerto serial

Aunque estas pruebas aun son simulaciones dado que se están realizando mediante el uso de una computadora constituyen un avance significativo debido a que la comunicación con el puerto serial se está realizando de manera correcta y el diseño y programación utilizado en esta parte es independiente de la aplicación que se desea realizar, en otras palabras la investigación y desarrollo realizado hasta este punto constituyen una tecnología que se puede utilizar para comunicarse con cualquier interfaz serial (ver apéndice de código).

Figura 4.11 Obtención de datos a partir de un botón programado en un formulario

4.5 Pruebas de comunicación con la Máquina de Hemodiálisis

Siguiendo con la secuencia de pruebas realizadas en el presente desarrollo damos paso a lo que vendría a ser la implementación del proyecto en la máquina de hemodiálisis, para ello es necesario utilizar una máquina real las cuales estuvieron disponibles para este fin en el Instituto de Ciencias Medicas y Nutrición Salvador Zubirán, en el área de Nefrología donde se utilizó una máquina de hemodiálisis modelo 2008H de la marca Fresenius, en esta parte se conectó la tarjeta de desarrollo a la interfaz serial de la máquina de Hemodiálisis y a su vez a una red local a fin de probar la interfaz web, tal como se muestra en la **figura 4.12**

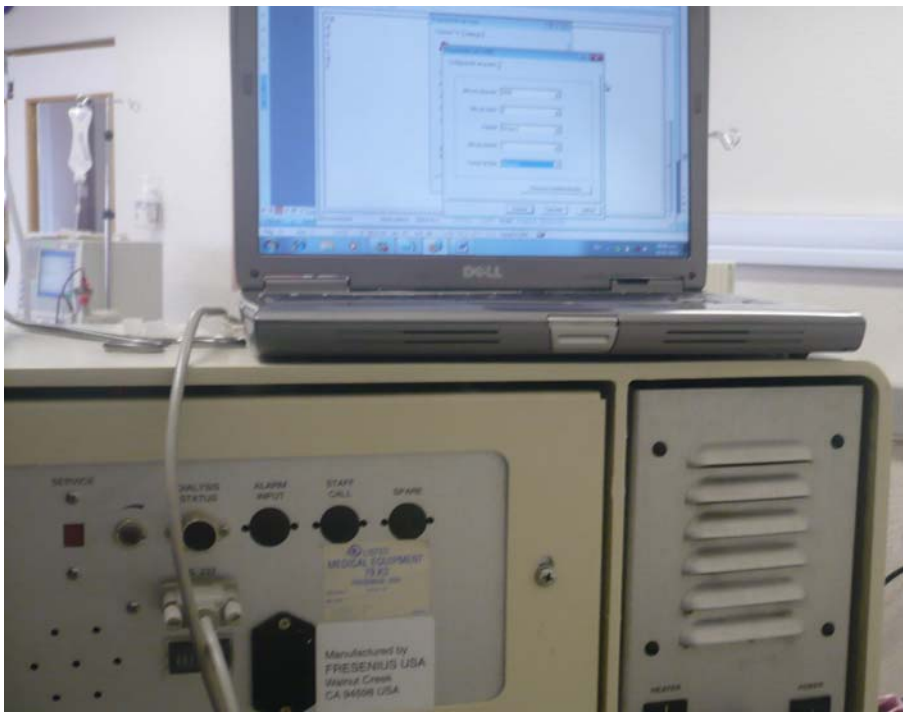


Figura 4.12 Comunicación con la máquina de Hemodiálisis

En esta parte se tuvo el problema de que al enviar el parámetro desde la página web a la tarjeta de desarrollo y esta realizar la lectura del puerto serial no se obtenía ningún valor y pareciera que no había ninguna respuesta por parte de la máquina de hemodiálisis, con el fin de verificar si la máquina estaba recibiendo o enviando valores a partir del puerto serial se conecto directamente la computadora a la interfaz serial de la máquina con la siguiente configuración en la hyperterminal.

Velocidad de Transferencia 9600 (bps), modo de configuración de 8 bits sin bit de paridad y en modo asíncrono tal como podemos apreciar en la **figura 4.13,** esta configuración de acuerdo al manual de usuario de la máquina de hemodiálisis Fresenius modelo 2008H **[2]**

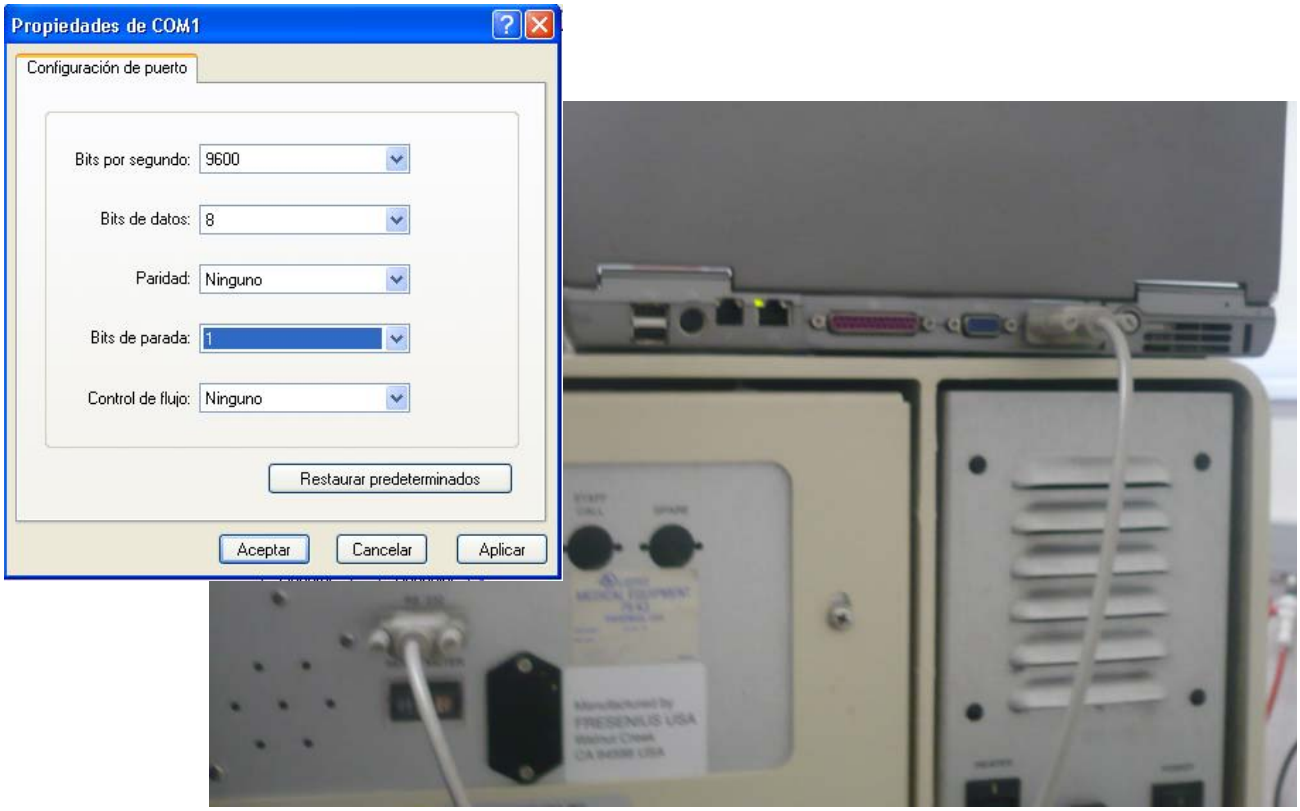


Figura 4.13 Verificación de error de envió de datos

Sin embargo nos enfrentamos al problema de que aun así no se recibió ninguna respuesta por parte de la máquina, por lo que se procedió a utilizar la interfaz de comunicación realizada en la tesis **[4],** con el fin de verificar el funcionamiento de la interfaz serial, utilizando una aplicación programada en MATLAB® y con comunicación al puerto serial se logro verificar que la máquina de hemodiálisis respondía correctamente como se muestra en la siguiente **figura 4.14**

Una vez que se verificó que la máquina funcionaba correctamente fué necesario corregir la aplicación en la tarjeta de desarrollo, para ello basándonos en la aplicación de MATLAB® y en el manual de usuario de la máquina se encontró que para que la máquina de hemodiálisis responda con los valores se le deben enviar los siguientes comandos:

- VP Presión Venosa
- AP Presión arterial
- TM Presión Trans-Membrenal
- TP Temperatura
- DF Flujo Dializante
- CD Conductividad
- BF Flujo Sanguíneo

- UR Rango de ultrafiltrado
- UT Estado de ultrafiltrado
- SY Presión Sistólica
- DI Presión Diastólica
- PL Pulso
- AL Alarmas
- CX Borrado de la interfaz y parado de la transferencia de datos.

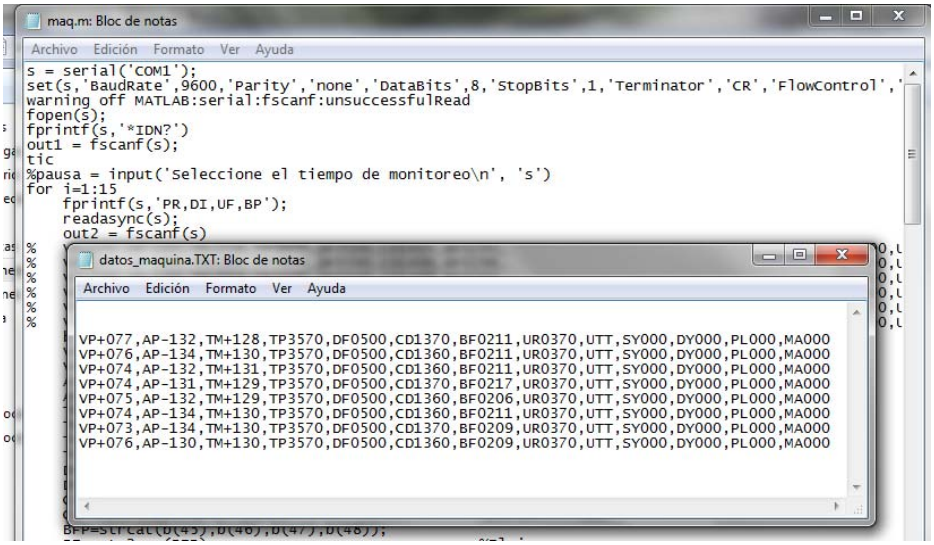


Figura 4.14 Resultados obtenidos de comunicación serial con MATLAB

Ya que se corrigió lo anterior en la tarjeta de desarrollo se procedió nuevamente a conectar la tarjeta con la máquina y la computadora vía red local, ahora cuando la página web envía el parámetro a la tarjeta mediante el formulario HTML, la tarjeta envía al puerto serial los comandos PR, DI, UF y BP como un cadena de texto y posteriormente se realiza la lectura del puerto serial en done esta vez sí se logro obtener los valores los cuales son enviados directamente a la página web y mostrados de forma automática tal como se muestra en la **figura 4.15**

4.6 Pruebas de Monitoreo de la Máquina de Hemodiálisis

Realizando correctamente la comunicación con la máquina de Hemodiálisis, se procede a verificar si las lecturas de los parámetros fisiológicos se realizan correctamente, para ello se implemento una aplicación en donde inicialmente se enviaba automáticamente un parámetro desde el formulario HTML a la tarjeta de desarrollo y esta a su vez realiza la lectura de la máquina de Hemodiálisis vía puerto serial y posteriormente mostrados en la página web, este procedimiento fue programado para que se realizara en un periodo de cada 5 minutos durante una hora y se logro observar que los valores en la página eran actualizados correctamente tal como se muestra en la **figura 4.16**

maq.m: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
s = serial('COM1');
set(s,'BaudRate',9600,'Parity','none','DataBits',8,'StopBits',1,'
warning off MATLAB:serial:fscanf:unsuccessfulRead
fopen(s);
fprintf(s,'*IDN?')
out1 = fscanf(s);
tic
%pausa = input('Seleccione el tiempo de monitoreo\n', 's')
for i=1:15
    fprintf(s,'PR,DI,UF,BP');
    readasync(s);
    out2 = fscanf(s)
%   VP+191,AP-185,TM+056,TP3600,DF0500,CD1400,BF0293,
%   VP+190,AP-188,TM+064,TP3600,DF0500,CD1400,BF0296,
```

Figura 4.15 Obtención de datos de la máquina de Hemodiálisis

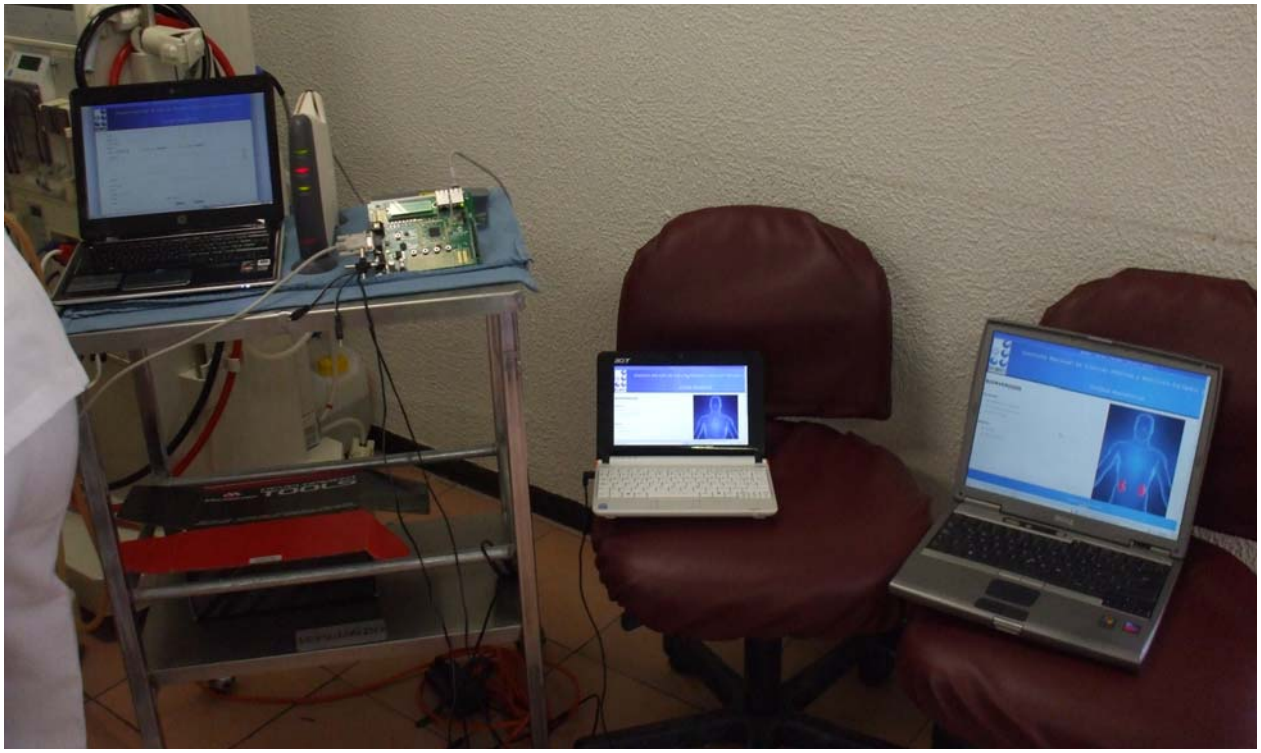


Figura 4.16 Monitoreo de datos

La siguiente implementación consistió en adicionar a la aplicación anterior un botón en un formulario HTML tal como se venía realizando anteriormente, esto se hizo con el fin que la lectura de los parámetros fisiológicos provenientes de la máquina de hemodiálisis se efectuara ya sea por lapsos de tiempo o en un momento deseado, sin que un método afecte a otro, esto se logro correctamente, sin embargo se pudieron apreciar casos en que las lecturas se hacían casi de manera simultánea, por

ejemplo si el botón del formulario HTML se presiona cuando el lapso de tiempo esta a punto de concluir la tarjeta de desarrollo realizaba las dos lecturas de manera seguida, lo que provocaba que los valores desplegados en la página web no se apreciaran correctamente, para solucionar este problema se modificó la aplicación de tal forma de que si se efectuaba la lectura de los parámetros fisiológicos mediante el botón del formulario HTML se reinicializaba el lapso de tiempo programado anteriormente, con el fin de evitar un la lectura simultanea (ver apéndice de código), tal como se muestra en la **figura 4.17.**

Figura 4.17 Funcionamiento correcto de la máquina de Hemodiálisis y la tarjeta de desarrollo

4.7 Pruebas de captura de datos hacia la BD

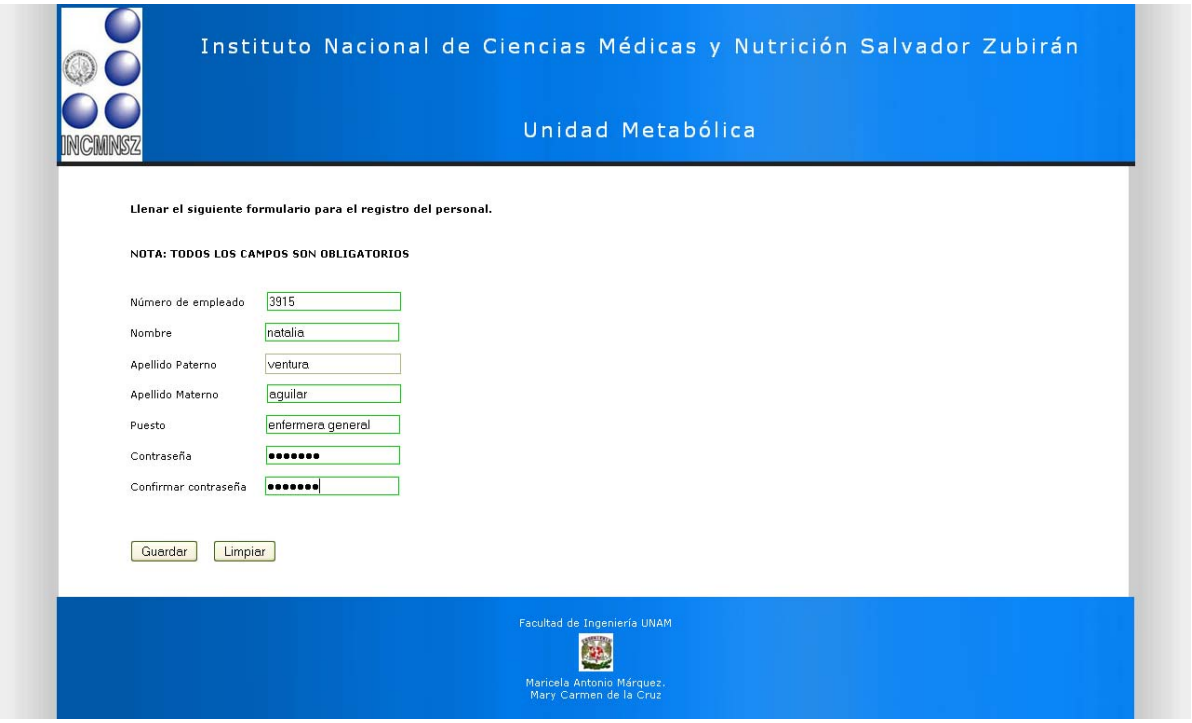
Parte importante de las pruebas del sistema consiste en corroborar que la información introducida, ya sea por el usuario como la obtenida a través de la máquina de hemodiálisis es correctamente guardada dentro de la base de datos.

Para ello se elaboraron pruebas a cada una de las ventanas introduciendo información y posteriormente realizar consultas sql a la base para confirmar que la información introducida fue almacenada dentro de la base de datos.

En cada una de las pantallas utilizadas en el sistema se utilizó una validación de datos, lo anterior con el objetivo de no introducir valores erróneos, es decir, que no corresponda la información (ejemplo: colocar teléfono donde se requiere un nombre) o que un campo que es obligatorio se encuentre vacío. Lo anterior se realizó con ayuda de java script a través de livevalidation.

Dentro de la programación se incluyó código para que las cadenas que se introduzcan a la base de datos se almacenen con letras mayúsculas y con ello exista una mayor uniformidad.

A continuación se muestra, algunas pruebas realizadas junto con la verificación de la información almacenada.



id_empleado	nombre	ap_paterno	ap_materno	puesto	contrasena
2954	MARICELA	ANTONIO	MARQUEZ	ENFERMERA	2819b146d382d0c89547e486f4250859
3915	NATALIA	VENTURA	AGUILAR	ENFERMERA GENERAL	2819b146d382d0c89547e486f4250859
123456	MIGUEL ANGEL	LOPEZ	CASTILLO	ENFERMERO	98a8d3f11b400ddc06d7343375b71a84
345678	MARY CARMEN	DE LA CRUZ	CRUZ	INGENIERA	e10adc3949ba59abbe56e057f20f883e

Figura 4.18 Prueba registro de personal

En la parte inferior de la Figura 4.18 se puede observar que el empleado fue registrado en la base, además es importante destacar que el campo de contraseña está protegida por el algoritmo de reducción criptográfica MD5.

En la parte de registro de personal, el campo que necesitó un mayor cuidado durante el diseño de la base de datos fue el de contraseña. Como ya se mencionó anteriormente, en este campo se utilizó MD5 por lo tanto se tomó en cuenta el mínimo de caracteres que el usuario tiene que introducir, así como un máximo que se le permita, ya que el usuario puede elegir una contraseña demasiado larga y no guardarse completamente al no contemplar la longitud del campo de contraseña a consecuencia de esto si no se guardaba en su totalidad en futuras ocasiones el empleado no tendría acceso al sistema.

El problema anterior se resolvió acotando el mínimo y máximo de caracteres para la contraseña, además de tener una longitud grande para el campo en cuestión.

- Registro pacientes

En la sección de registro de pacientes (Figura 4.19) se llena la información del paciente que recién ingresa al área para recibir su primera diálisis.

El problema a tratar dentro de este apartado del sistema es el formato de las fechas (fecha de ingreso y fecha de nacimiento). Para los campos de fechas en la base de datos se manejo el tipo DATE, el cual tiene el siguiente formato: ‘2009-12-31’.

Para evitar que el usuario introduzca las fechas con otro formato que no sea manejado por mysql, se colocó dentro de las cajas de textos el formato que se necesita tener dentro de este campo, además de estar validado para que no existan errores dentro del mismo.

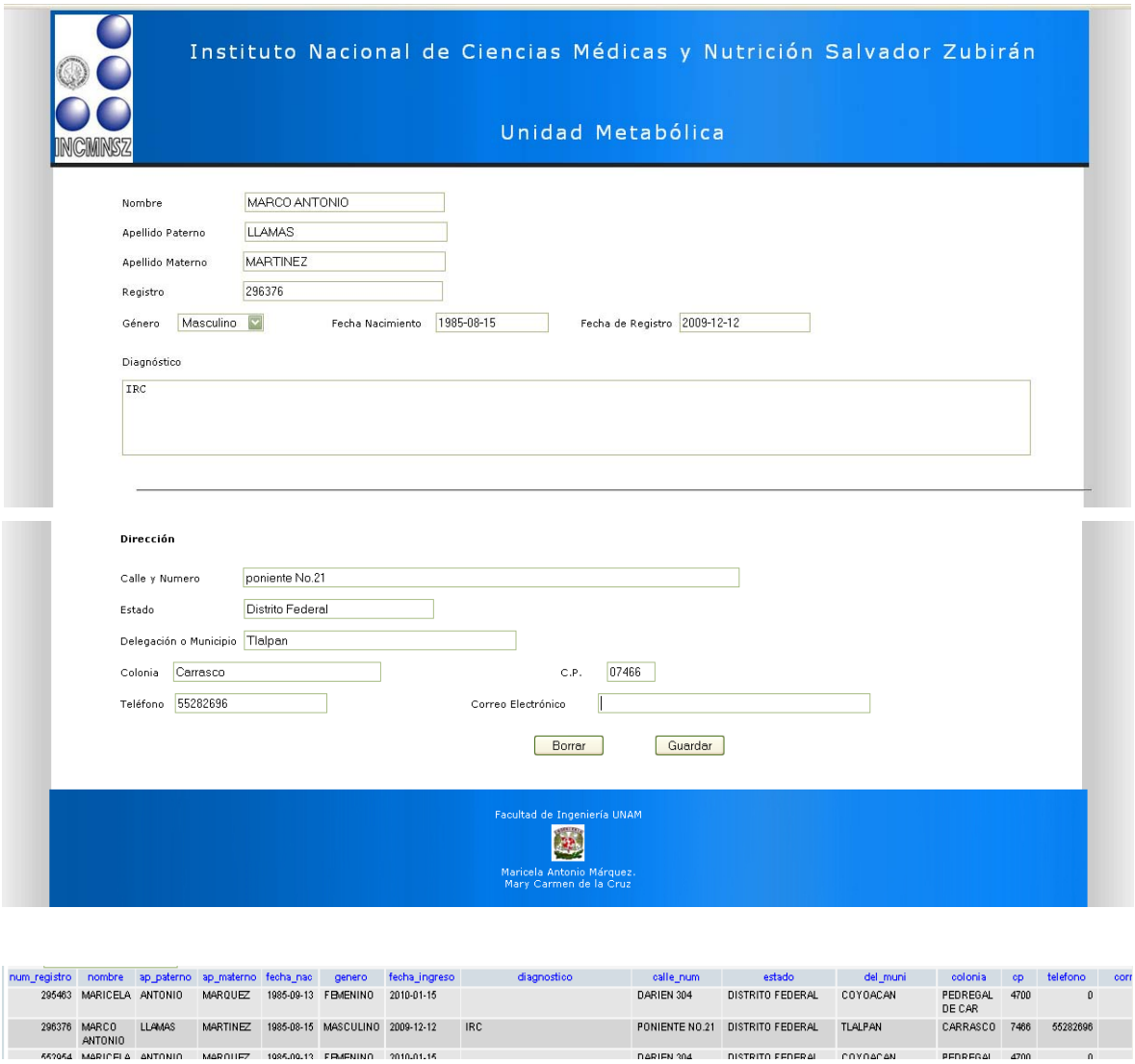
Una vez que la información es guardada el sistema lleva al usuario a la ventana de prediálisis para iniciar la sesión del paciente a quien se haya registrado.

- Prediálisis

En la parte de prediálisis del paciente ya se manejan datos de tipo entero, flotante, varchar, fecha, hora.

El principal elemento con el que se tuvo que tener cuidado fue en obtener la información de la base de datos tanto de la máquina que se está utilizando y obtener los datos del personal del área.

En la figura 4.20 se observa la etiqueta “Conecta”. Durante la diálisis es importante conocer quién es el responsable de iniciar la sesión del paciente, así dentro de la caja de selección se muestran los empleados registrados dentro de la base de datos. Lo anterior no fue problema ya que previamente se tiene la tabla con la información necesaria del personal y sólo se ingresa a esa información por medio de código PHP y una consulta por medio de SQL.



num_registro	nombre	ap_paterno	ap_materno	fecha_nac	genero	fecha_ingreso	diagnostico	calle_num	estado	del_muni	colonia	cp	telefono	corr
295463	MARICELA	ANTONIO	MARQUEZ	1985-09-13	FEMENINO	2010-01-15		DARIEN 304	DISTRITO FEDERAL	COYOACAN	PEDREGAL DE CAR	4700	0	
296376	MARCO ANTONIO	LLAMAS	MARTINEZ	1985-08-15	MASCULINO	2009-12-12	IRC	PONIENTE NO.21	DISTRITO FEDERAL	TLALPAN	CARRASCO	7466	55282696	
552954	MARICELA	ANTONIO	MARQUEZ	1985-09-13	FEMENINO	2010-01-15		DARIEN 304	DISTRITO FEDERAL	COYOACAN	PEDREGAL	4700	0	

Figura 4.19 Prueba Registro de pacientes

- Diálisis

Una de las partes más importantes de una sesión es el momento en que recibimos la información de la máquina de hemodiálisis. Esta información se encuentra en constante cambio (dependiendo de la evolución de la sesión y del estado del paciente) se tiene que almacenar cada determinado tiempo (30 minutos) por lo tanto es necesario llenar la plantilla con los datos necesarios.

Parte de la información proviene de la máquina de hemodiálisis; sin embargo, otros parámetros se tienen que obtener a través de otros instrumentos que no están conectados con la máquina y estos campos tienen que ser llenados manualmente por el usuario.

Con el fin de probar el correcto almacenamiento de la información dentro de la base de datos y el no poder contar en esos momentos con la máquina de hemodiálisis, se utilizó un archivo de texto con información aleatoria.

Figura 4.20 Prueba Llenado de prediálisis

Se programó de tal manera que por medio de la aplicación AJAX se invoque una sentencia que a su vez se comunica con archivo de texto que contiene valores similares a los provenientes de la máquina de hemodiálisis y automáticamente se llenaran los campos solicitados. El resultado se muestra en la siguiente figura 4.21.

Nota: La prueba para de esta sección con los datos arrojados por la máquina de hemodiálisis se encuentran ubicados en Pruebas al sistema completo.

id_dialisis	f_dial	p_art	p_ven	f_san	ptm	vol_sp	v_uf	ta	fc	f_resp	na_p	hb_hto	v_sang	acm	kt_v
1	333.75	364324	373	3826	2774	37464	9764	2746	55.89	2424	3475	2466	2047	3744	37450
2	333.75	364324	373	3826	2774	37464	9764	2746	55.89	2424	3475	2466	2047	3744	37450

Figura 4.21 Prueba Llenado de diálisis

- Postdiálisis

Como última prueba de captura de datos se tiene la ventana de postdiálisis. Para ello se introdujeron datos y se comprobaron posteriormente el almacenamiento dentro de la base de datos.

Para ésta sección los datos más importantes a verificar fue saber si correspondían a datos tipo flotante o tipo entero, así como checar se obtuviera correctamente los datos del personal que desconecta al paciente.

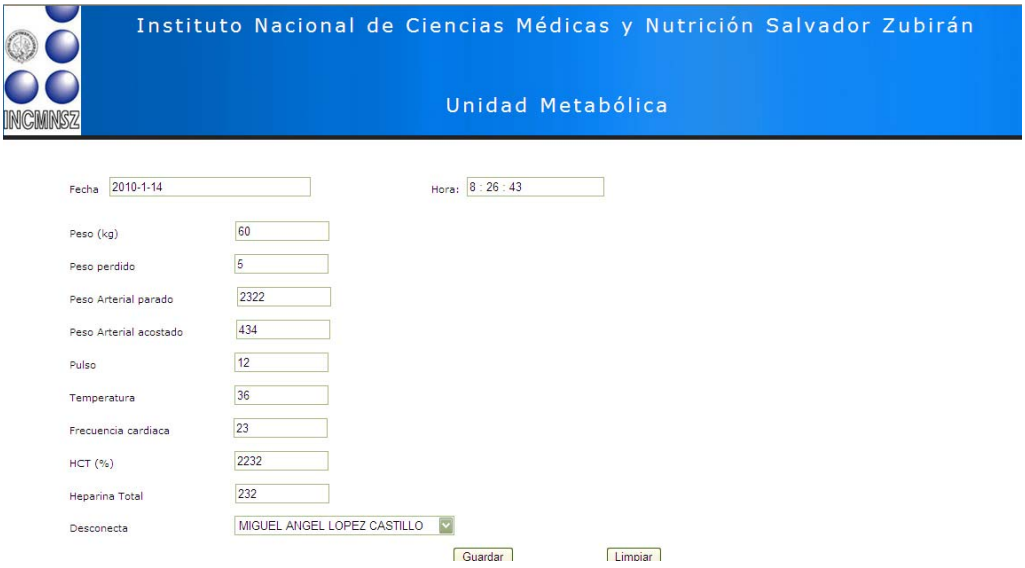


Figura 4.22 Prueba Llenado de postdiálisis

4.8 Pruebas de monitorio vía TCP/IP

Para realizar las pruebas de monitoreo vía TCP/IP fue necesario colocar una red local entre dos computadoras. Una funciona como el servidor, mientras la segunda es el cliente.

En la Figura 4.24 se muestra una imagen de las dos computadoras en funcionamiento. La computadora ubicada a la izquierda funge como el servidor, la IP del servidor es 192.168.1.64. La computadora de la derecha tiene la IP 192.168.1.15

Nota: Las direcciones IP del servidor como del cliente se han obtenido automáticamente, no se realizó ninguna configuración para las mismas.

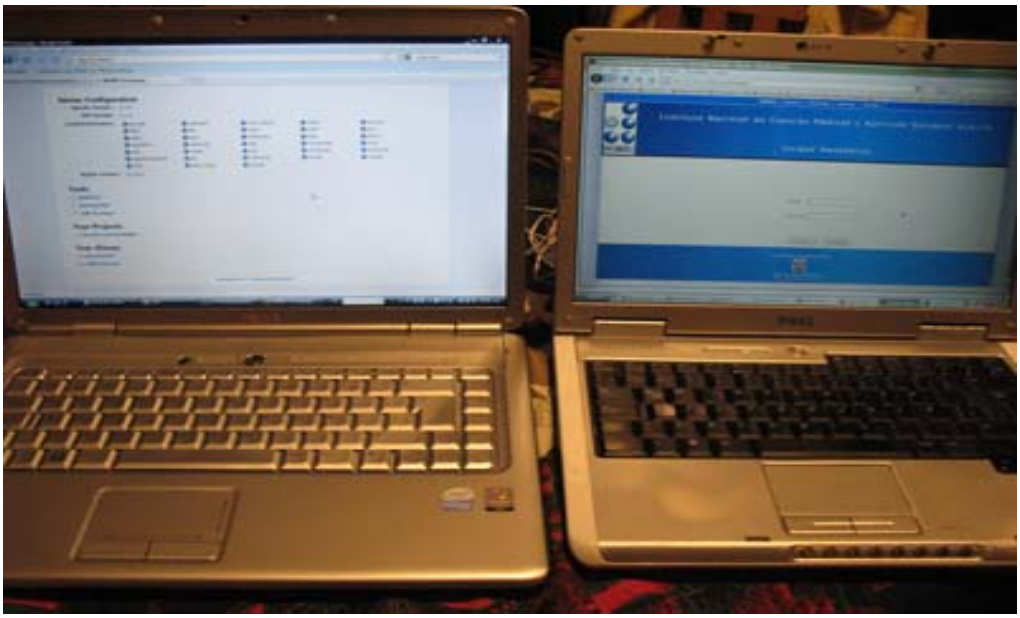


Figura 4.24 Prueba TCP/IP Cliente(derecha)-Servidor(izquierda)

El propósito de estas pruebas es verificar que la comunicación con la base de datos sea la correcta, además que las actualizaciones tengan lugar tal como se tenían planteadas.

A continuación se elaboran pruebas de comunicación con la computadora servidor y la computadora cliente para verificar el correcto acceso al sistema y su almacenamiento de datos.

Para realizar estas pruebas se utilizarán los registros que fueron almacenados como pruebas en el tema 4.7. Posteriormente se procede a entrar a la aplicación desde la computadora cliente, colocando en el navegador de internet la dirección del servidor.

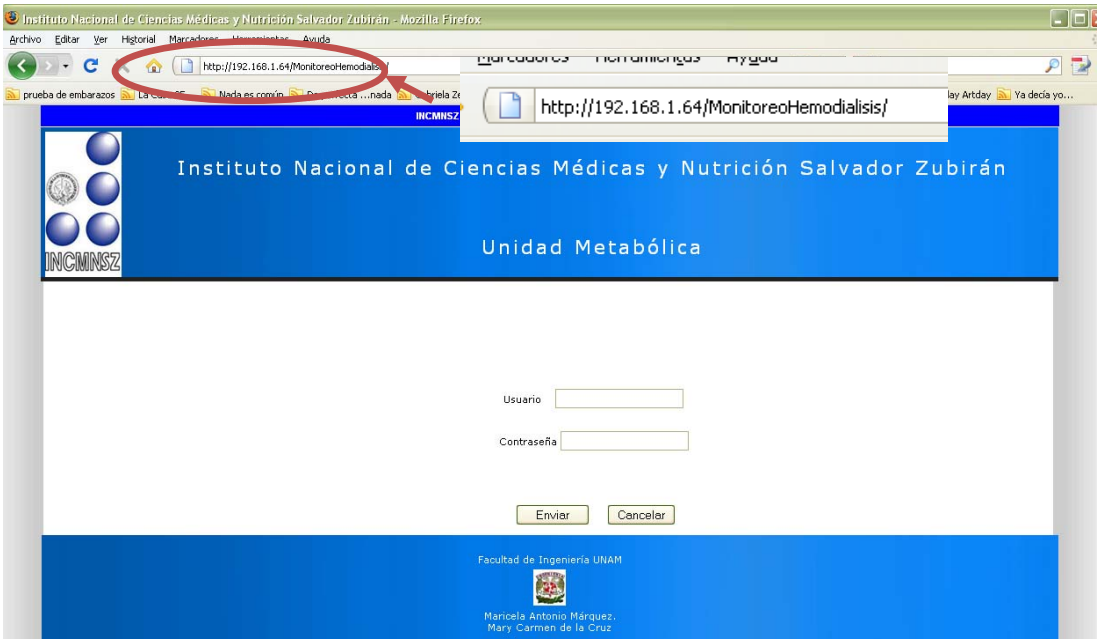


Figura 4.25 Accesando a la aplicación desde modo cliente

El siguiente paso es iniciar sesión utilizando como Usuario el identificador '3915', la contraseña utilizada fue aquella que se introdujo al ser registrado el usuario en la etapa de pruebas

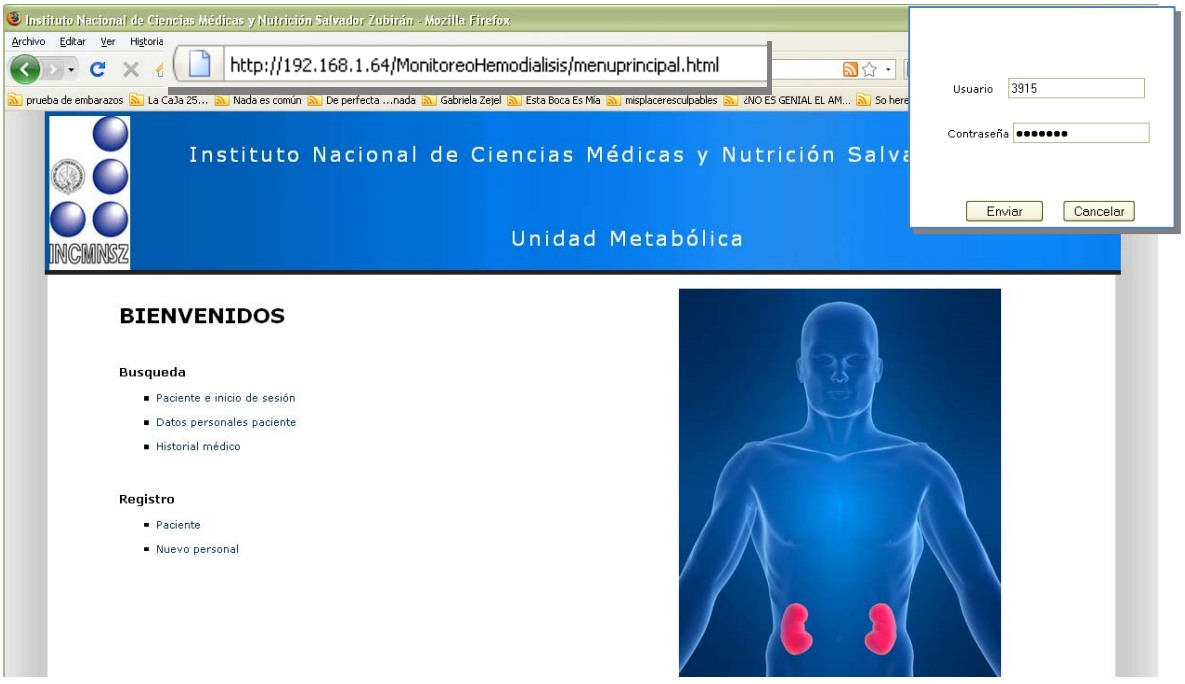


Figura 4.26 Acceso Menú principal desde modo cliente

Para mostrar que el sistema funciona se introdujo una contraseña incorrecta, con lo cual no se tuvo acceso, al colocar la contraseña correcta el usuario es direccionado al menú principal.

La próxima prueba consiste en registrar a un nuevo paciente desde la computadora que funge como cliente y verificar que se ha guardado dicha información desde nuestro servidor.

En la figura 4.27 se muestra en la parte de atrás como se introduce la información en la computadora cliente y en la parte delantera se muestra que la información ha sido almacenada en el servidor.

Las pruebas de registro (desde el cliente) y verificación de almacenamiento (desde el servidor) se aplicaron a todas las ventanas que componen nuestro sistema (registro personal, prediálisis, diálisis, postdiálisis) obteniendo como resultado una correcta comunicación remota entre la máquina cliente y la máquina servidor.

Durante la realización de las pruebas vía TCP/IP se verificó que el servidor estuviera en funcionamiento todo el tiempo ya que si en algún momento nuestro servidor dejara de funcionar, no se podía guardar ni consultar la información que se estaba introduciendo o consultando.

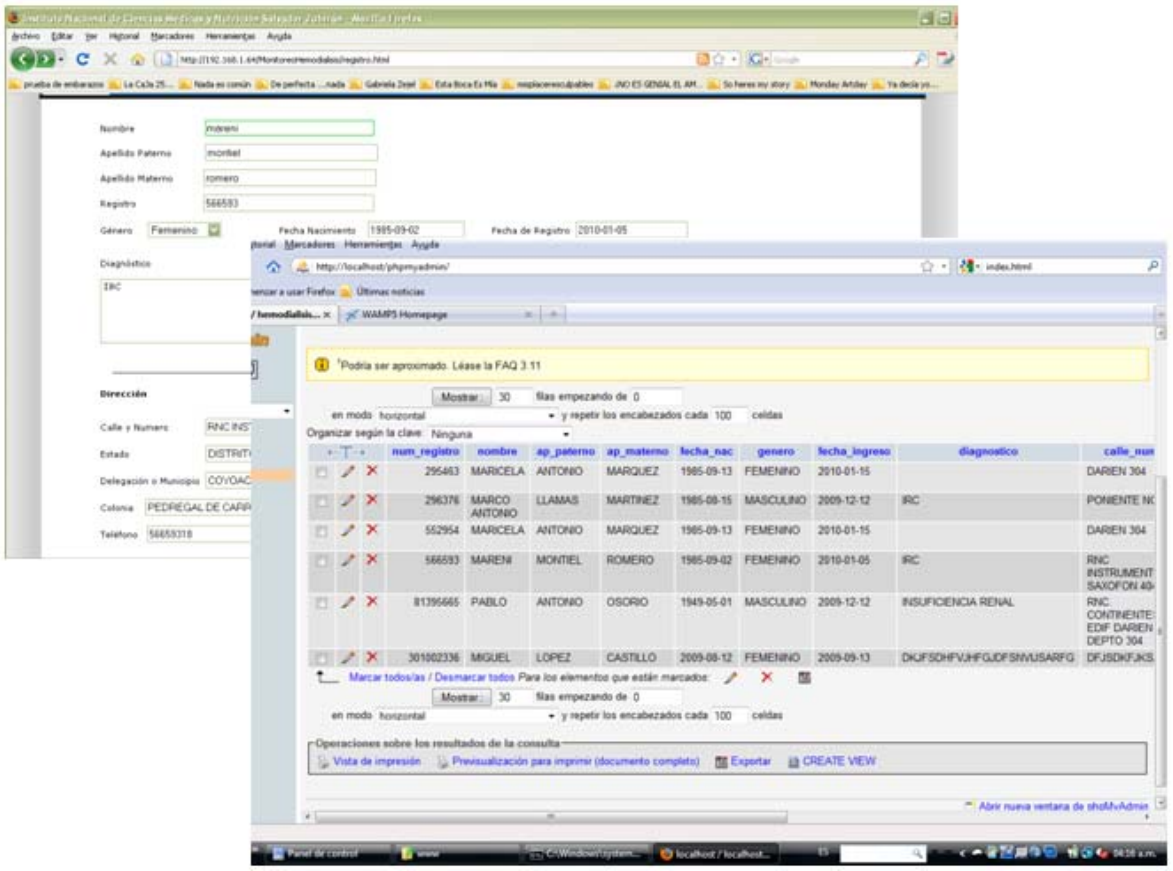


Figura 4.27 Almacenando paciente desde modo cliente

Con la prueba anterior se demostró que la comunicación remota, establecida entre la máquina cliente y la máquina servidor es correcta.

Como última prueba a demostrar en este documento a continuación se muestra otra prueba de almacenamiento de datos, para este caso se mostrará la correcta inserción de datos para postdiálisis.

4.9 Prueba de la Interfaz Web

Este tipo de pruebas tiene como finalidad:

- Verificar que tan fácil de usar es el sistema,
- Verificar el aprendizaje, es decir que tan fácil es para el usuario realizar tareas básicas la primera vez que entra en contacto con el sistema.
- Evaluar que tan rápido los usuarios llevan a cabo las tareas una vez que han aprendido algo de él.
- Conocer que tan satisfactorio es usar el sistema

Estas pruebas se elaboraron durante todo el proyecto ya que se necesitó de la colaboración del personal para conocer los datos a manejar para el proyecto, para ello, se elaboraron entrevistas previas para conocer las características de cada uno de los datos, es decir, si era un valor tipo entero o flotante,

Debido a que existió una comunicación con los usuarios finales (principalmente enfermeras) se integró en el sistema toda la información que se llenaba manualmente de acuerdo a formatos manejados por ellos; sin embargo, nuestro usuario no utilizaba todos los rubros colocados dentro de las mismas y actualmente utilizan otra información que no está agregada, por lo tanto se prosiguió a eliminar y agregar información solicitada por nuestros usuarios principales.

De esta manera cada vez que se agregaban elementos al sistema se mostraba a nuestro usuario para que existiera una realimentación y se contara con todos los datos necesarios, en un formato que entendiera el personal que utilizará el producto final y eliminando todo lo que no fuera necesario para evitar confusiones.

Como prueba final se le pidió a un usuario final, en este caso a un enfermero, que elabora una prueba del sistema. Sin previa explicación del funcionamiento de la interfaz, se le solicito que entrara al sistema,

Cabe aclarar que no se dio explicación alguna al usuario debido a que es importante que caigan en errores y así mejorar el sistema en aquellos casos donde se presentaron fallos. El único dato dado a ellos fue un usuario y contraseña para ingresar al sistema.

Figura 4.28 Usuario utilizando el sistema

Aunque ya se habían llevado a cabo entrevistas con diferentes personas que utilizarán el sistema, el usuario que efectuó la prueba no había sido entrevistado con anterioridad por lo tanto no estaba empapado de toda la información solicitada a sus compañeros y por supuesto contó con un diferente punto de vista. De esta manera, realizó algunos comentarios a agregar al sistema pues lo considera útil de acuerdo a la experiencia que tiene trabajando en esta área del hospital y con la finalidad de mejorar el proyecto desarrollado.

En general comentó ser una herramienta fácil de utilizar, concisa y bastante útil para el desarrollo de su trabajo. Por lo tanto, al tener una la satisfacción del usuario quien puso a prueba el sistema se concluye que se obtuvo exitosamente un sistema que cumplía los requerimientos de nuestro usuario.

4.10 Pruebas al sistema completo

Las pruebas finales se realizaron al conectar la tarjeta de desarrollo PICDEM NET2 a la máquina de hemodiálisis, cuyo modelo es 2008H de la compañía Fresenius, que previamente fue configurada con una IP estática.

Posteriormente se comprobó que existía comunicación del servidor (tarjeta de desarrollo) con la máquina de hemodiálisis y con la computadora central. El servidor y la computadora que monitorea están conectadas a través de una red local creada por medio de un ruteador.

En la computadora central se inicia sesión para entrar al sistema, posteriormente se registra o busca al paciente (dependiendo del caso) que se conectará a la máquina de hemodiálisis.

De acuerdo al procedimiento que se realiza para una sesión de diálisis, una vez registrado el paciente se procede a llenar los parámetros de prediálisis. Al llenar esta información se inicia la sesión de diálisis en la cual la computadora de monitoreo ingresa por medio de la red a la tarjeta de desarrollo (PICDEM NET2), la cual contiene los parámetros fisiológicos (presión transmembrana, presión arterial, presión venosa, etc.) que va obteniendo la máquina de hemodiálisis del paciente. Este proceso de lectura y almacenamiento de parámetros se realiza durante toda la sesión de diálisis, la cual, como ya se ha informado llega a durar hasta 4 horas. Al terminar la sesión se procede a llenar la información de postdiálisis y posteriormente se termina el procedimiento.

Toda la información recabada durante la sesión puede ser consultada en el momento en que el usuario así lo requiera.

Figura 4.29 Prueba de funcionamiento de todo el sistema

4.11 Resultados Obtenidos

Como resultado de la investigación y desarrollo de dicho proyecto se obtuvo:

A partir de nuevas tecnologías como lo fue un microcontrolador donde se alberga una página que almacenan los datos provenientes de una máquina de hemodiálisis y estos a su vez manipulados a través de páginas dinámicas.

La comunicación de un microcontrolador y la interfaz serial de una máquina de hemodiálisis lo cual nos satisface pues comprendimos el funcionamiento de tal forma que no solo se puede aplicar este tipo de comunicación a las máquinas de hemodiálisis sino a cualquier equipo que posea una interfaz serial y que en el ámbito hospitalario siguen estando presente en muchos equipos de uso biomédico.

Un diseño en programación con código en C18 el cual es un código abierto y disponible para cualquier persona, para la manipulación del puerto serial e interpretación de los datos provenientes de la máquina de hemodiálisis.

Comprensión y manipulación del funcionamiento de la Microchip TCP/IP para el diseño de un programa que es capaz de comunicarse con cualquier equipo que posea puerto serial.

Una interfaz web concisa que muestra los valores primordiales para el personal que monitorea de forma remota los parámetros fisiológicos de los pacientes.

El sistema información está constituido una base de datos que tiene integridad total, es decir, confiable, atómica, estable y segura, ya que cumple con reglas de normalización (3N), por lo tanto no permite alteraciones no previstas a la misma.

El sistema ha sido desarrollado para pocas máquinas de hemodiálisis y una computadora personal tiene el funcionamiento de servidor; sin embargo, con el desarrollo de este proyecto se establecen las bases para un sistema más robusto. Por lo tanto se puede realizar en un futuro una migración a un servidor dedicado, la expansión del sistema a más máquinas de hemodiálisis y una base de datos más grande.

Uniendo los conocimientos de computación y la biomédica y haciendo uso de nuevas tecnologías se logro el desarrollo de un sistema computacional capaz de realizar el monitoreo remoto de parámetros fisiológicos que proporciona una máquina de Hemodiálisis, dando así las bases para poder implementarlo en cualquier dispositivo que cuente con una interfaz RS-232 y que sea de ayuda para la comunidad hospitalaria.

CAPÍTULO V

CONCLUSIONES Y TRABAJO A FUTURO

CAPÍTULO V CONCLUSIONES Y TRABAJO A FUTURO

5.1 Conclusiones del Desarrollo del proyecto

Hoy en día la biomédica juega un papel muy importante dentro del campo de la medicina pues comprende los principios y técnicas de la ingeniería donde se mezclan carreras como Electrónica, Computación y Medicina, es por eso que el presente proyecto hace una aportación para la mejora de los procesos que tienen que ver con la transmisión de información biomédica.

El presente proyecto ha contribuido de manera importante no solo para reafirmar sino también para aplicar los conocimientos adquiridos durante una larga investigación, donde se aprendió de las características de nueva tecnología en microcontroladores, la gran utilidad que proporcionan los controladores Ethernet y el funcionamiento que tienen en conjunto.

Así mismo fue de gran importante adquirir los conocimientos en materia de fisiología, como de las funciones que desempeñan los riñones y los padecimientos que sufren los pacientes para poder entender el funcionamiento del sistema de la máquina de hemodiálisis.

La transmisión de datos de la central de monitoreo para máquinas de hemodiálisis Fresenius 2008H fue posible gracias a un sistema que integro todos los elementos necesarios para la conectividad a internet y al software de la Microchip Stack TCP/IP que se nos fue proporcionada y modificada para los fines propuestos.

Gracias a la aplicación de la computación como base principal de dicho proyecto, fue posible el diseño de una página web que unió tecnologías como HTML, PHP y AJAX en la cual se muestra de manera concisa a través de una interfaz gráfica la información extraída de la máquina de hemodiálisis.

Para finalizar la importancia que nuestro proyecto ha destacado es que aunque existen equipos como simuladores y sistemas expertos que hacen el mismo tipo de monitoreo y que pertenecen a múltiples compañías que se dedican a la industria biomédica, estos son de alto costo, que aunque dichas compañías ofrecen múltiples alternativas para la gente con necesidad de una dosis de diálisis no son tan fáciles de adquirir pues las máquinas por mencionar son altamente costosas y por ende el precio de una sesión es también de alto costo.

La biomédica sin embargo aun es una ciencia muy joven y por el momento en cuestiones computacionales se puede explotar más. Aunque nuestro proyecto no se desarrolla un equipo como tal biomédico nos dio la satisfacción de saber que sirve como base para otros trabajos que tengan el fin de

dar un mejor y más rápido diagnóstico de acuerdo a las necesidades que existan dentro del ámbito hospitalario.

5.2 Propuesta de Mejora

La ventaja que tiene nuestro proyecto con respecto a los ya existentes consiste en tener acceso a la información remotamente.

Como todo trabajo de investigación el nuestro queda abierto a mejoras y a nuevas aportaciones que pudieran complementar el presente desarrollo, a continuación ofrecemos algunas recomendaciones que sirven como trabajo a futuro:

- Diseño de una tarjeta con los dispositivos básicos.
- Monitorización de diferentes modelos de máquinas de hemodiálisis.
- Un estudio de Mercado donde se lleve a cabo el análisis de precios y comercialización.
- Tener un servidor de aplicaciones dedicado con un IP fija a fin de poder tener el monitoreo con salida a internet.
- Elaboración de estadísticas de la evolución del estado de los pacientes.
- Generación de reportes.
- Aplicar minería de datos para clasificar a los pacientes y tener un tratamiento predeterminado de acuerdo a sus características.

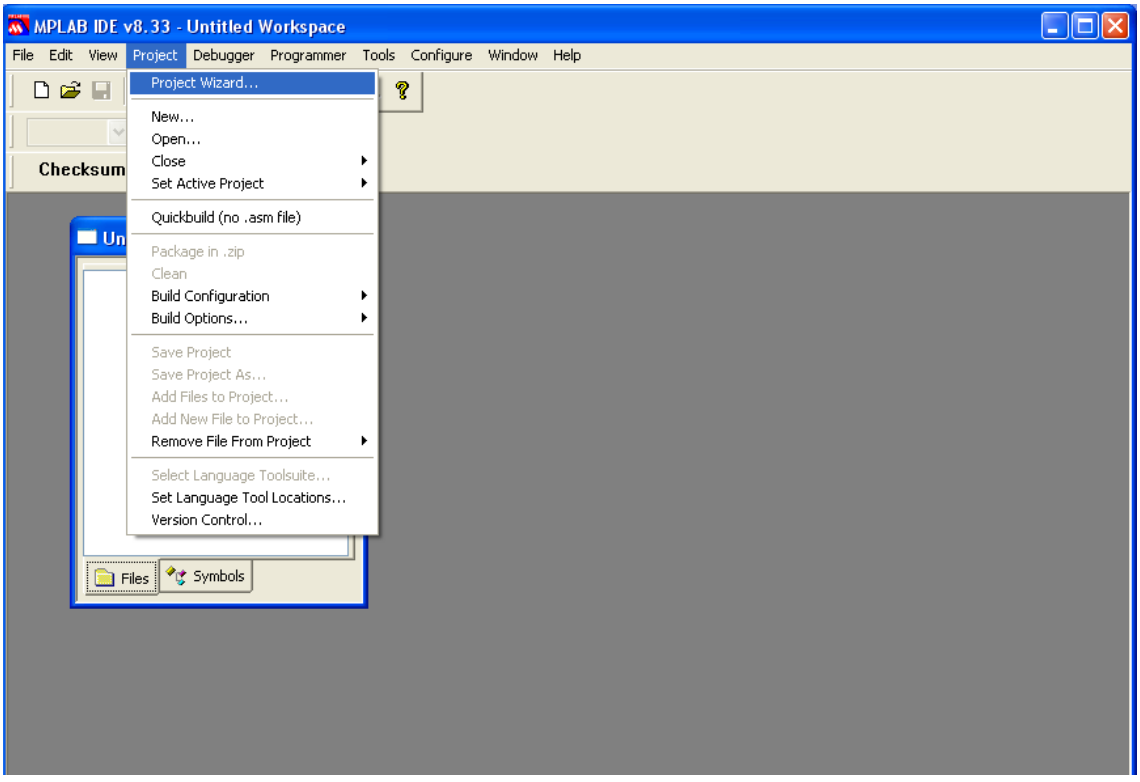
ANEXOS

Creación de proyectos en MPLAP

Para escribir un código y poder almacenarlo en el Microcontrolador de la Tarjeta PICDEM NET2 se realizan los siguientes pasos:

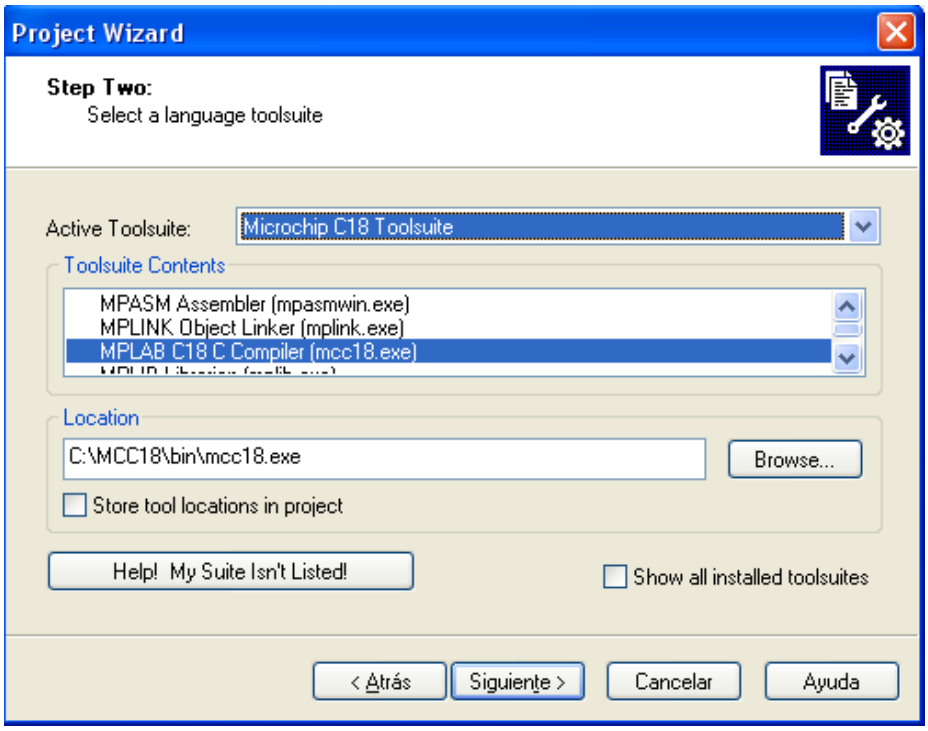
1. El código fuente puede ser escrito en cualquier editor de texto y debe guardarse con la extensión **.c**
2. Una vez que se instala MPLAB-C18 y se ha escrito el programa se crea un nuevo proyecto para poder compilar, simular, depurar y programar.

   Escoger la opción ***Proyect/Proyect Wizard*** y seguir las instrucciones que vayan apareciendo.
   - o Escoger el Microcontrolador (en nuestro caso PIC18F97J60)
   - o Escoger el conjunto de lenguajes Microchip toolsuite (en nuestro caso C18)

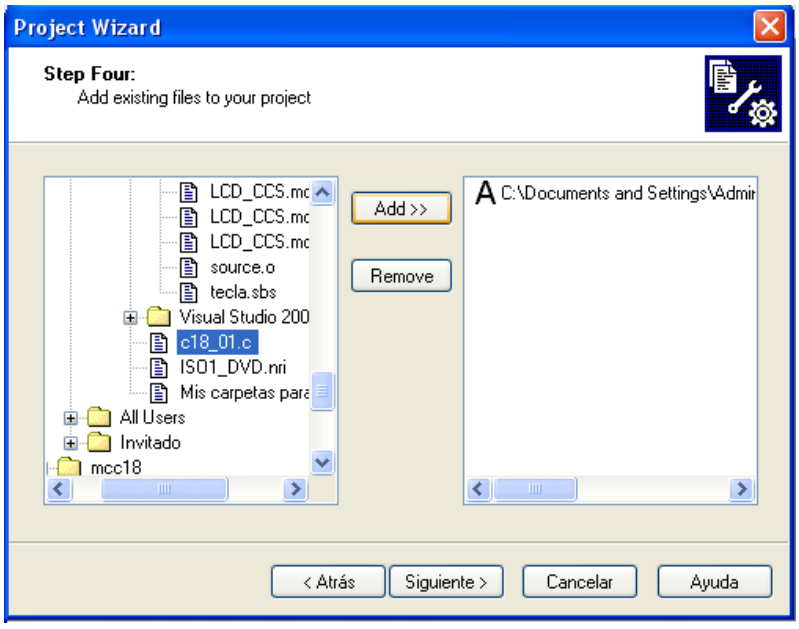


- Es recomendable poner el mismo nombre al archivo fuente que al nombre del proyecto

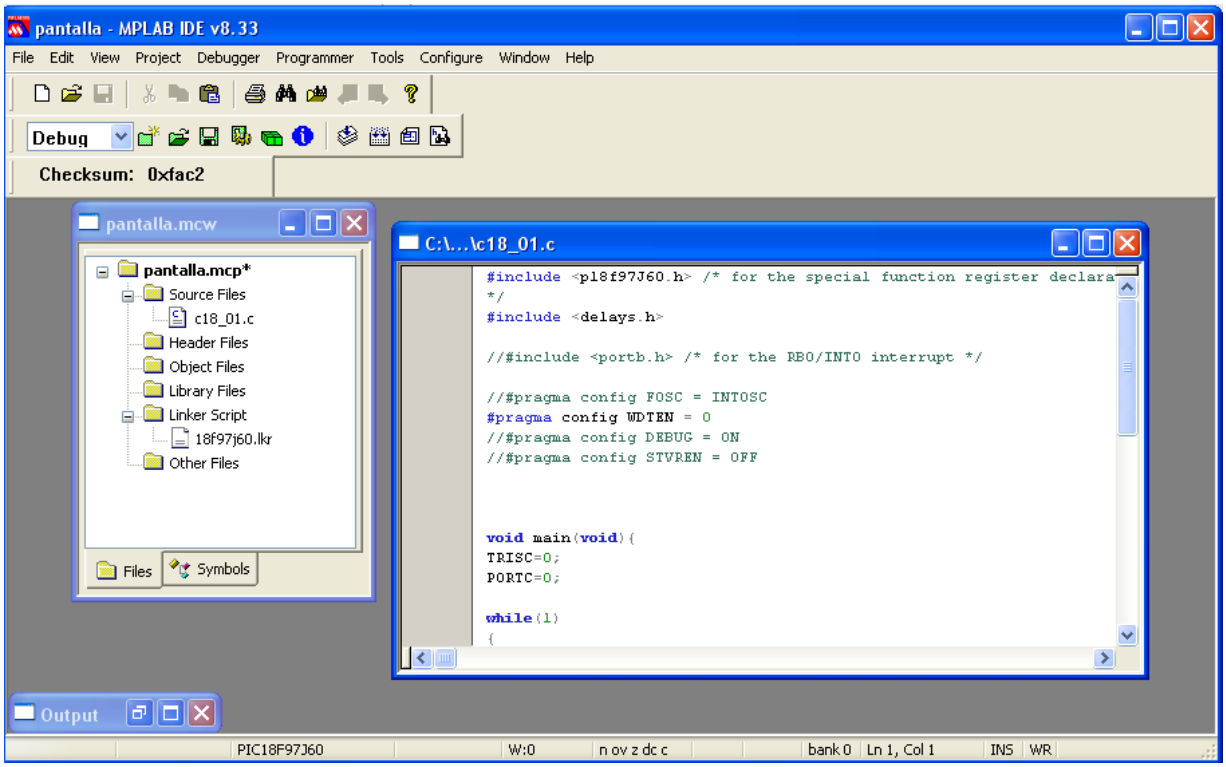
- Escoger el directorio donde se desea trabajar



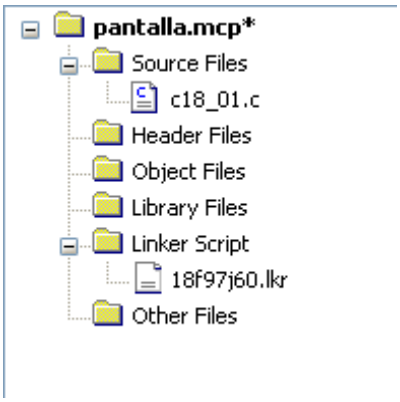
- A continuación seleccione el archivo fuente y dar ***Add***



- Al finalizar se habrá creado el proyecto
- Según la versión también se debe agregar al proyecto el archivo (PIC18F utilizado).lkr ubicado en MCC18/lkr, sino produce error de compilación.



- Una vez creado el proyecto, haga click en el ícono ***Built All***  . Con esto compilará el programa y se crearán archivos de error, mapa del programa, archivos objetos y archivos hex.
- Si no se han cometido errores al introducir los códigos aparecerá un mensaje que nos indicara que el programa se ha ensamblado con éxito

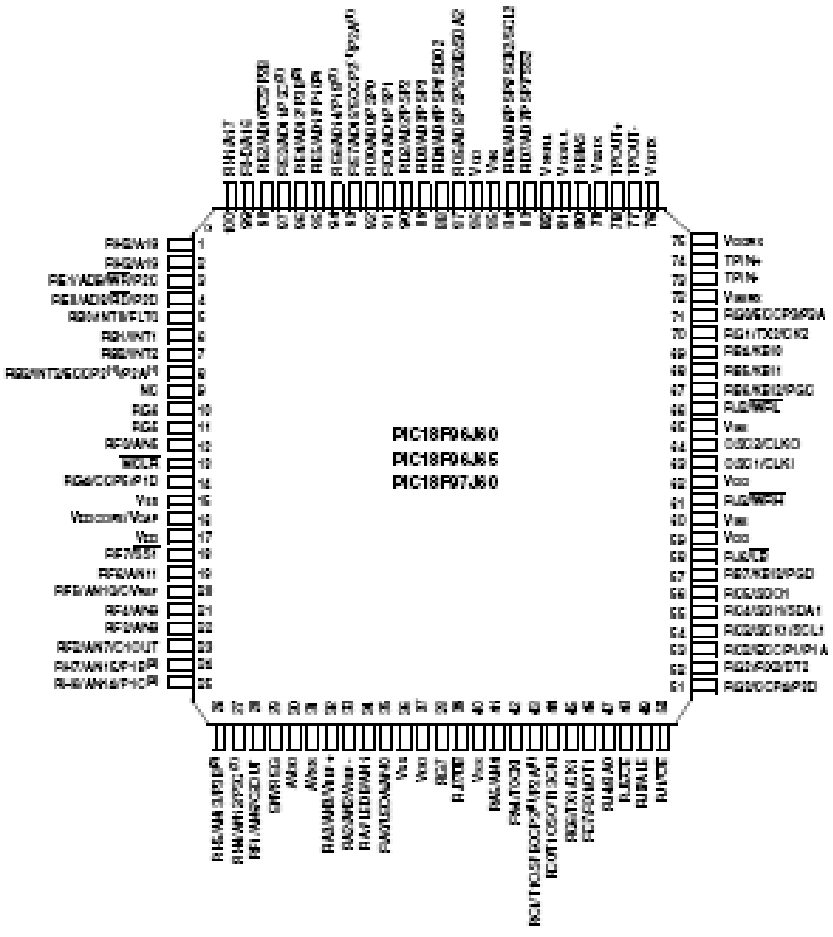


HOJA DE ESPECIFICACIONES DE LOS DISPOSITIVOS UTILIZADAS

PIC18F97J60 FAMILY

Pin Diagrams (Continued)

100-Pin TQFP



Note 1: The ECCP2/P2A pin placement depends on the CCP2MX Configuration bit and Processor mode settings.
2: P1B, P1C, P3B and P3C pin placement depends on the ECCPMX Configuration bit setting.

MICROCHIP

ENC28J60

Stand-Alone Ethernet Controller with SPI Interface

Ethernet Controller Features

- IEEE 802.3 compatible Ethernet controller
- Integrated MAC and 10BASE-T PHY
- Supports one 10BASE-T port with automatic polarity detection and correction
- Supports Full and Half-Duplex modes
- Programmable automatic retransmit on collision
- Programmable padding and CRC generation
- Programmable automatic rejection of erroneous packets
- SPI Interface with clock speeds up to 20 MHz

Buffer

- 8-Kbyte transmit/receive packet dual port SRAM
- Configurable transmit/receive buffer size
- Hardware-managed circular receive FIFO
- Byte-wide random and sequential access with auto-increment
- Internal DMA for fast data movement
- Hardware assisted checksum calculation for various network protocols

Medium Access Controller (MAC) Features

- Supports Unicast, Multicast and Broadcast packets
- Programmable receive packet filtering and wake-up host on logical AND or OR of the following:
  - Unicast destination address
  - Multicast address
  - Broadcast address
  - Magic Packet™
  - Group destination addresses as defined by 64-bit hash table
  - Programmable pattern matching of up to 64 bytes at user-defined offset

Physical Layer (PHY) Features

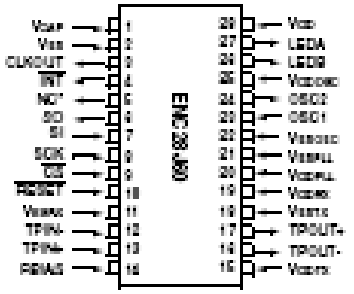
- Loopback mode
- Two programmable LED outputs for LINK, TX, RX, collision and full/half-duplex status

Operational

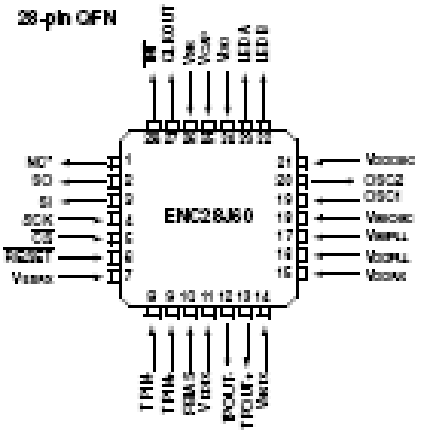
- Six interrupt sources and one interrupt output pin
- 25 MHz clock input requirement
- Clock out pin with programmable prescaler
- Operating voltage of 3.1V to 3.6V (3.3V typical)
- 5V tolerant inputs
- Temperature range: -40°C to +85°C Industrial, 0°C to +70°C Commercial (SSOP only)
- 28-pin SPDIP, SSOP, SOIC, QFN packages

Package Types

28-Pin SPDIP, SSOP, SOIC



28-pin QFN



* Reserved pin; always leave disconnected.

APÉNDICES
CÓDIGO DE APLICACIÓN EN EL MICROCONTROLADOR

El este apéndice se muestra el código fuente del programa principal para el funcionamiento de la tarjeta PICDEM.net 2

```
void main(void)
#endif
{
   static TICK t = 0;
          static TICK t1 = 0;

   // Initialize any application specific hardware.
   InitializeBoard();

#ifdef USE_LCD
// Initialize and display the stack version on the
LCDLCDInit();
DelayMs(100);
strcpypgm2ram(LCDText, "TCPStack " VERSION
" " "              ");
//strcpypgm2ram(LCDText,                "Valor"
VAR_MAC_ADDRESS );
          LCDUpdate();
#endif
 // Initialize all stack related components.
 // Following steps must be performed for all
applications using
   // PICmicro TCP/IP Stack.
   TickInit();

   // Following steps must be performed for all
applications using
   // PICmicro TCP/IP Stack.
   MPFSInit();

          // Load the default NetBIOS Host Name
memcpypgm2ram(AppConfig.NetBIOSName,
(ROM void*)MY_DEFAULT_HOST_NAME, 16);
FormatNetBIOSName(AppConfig.NetBIOSName);

// Initialize Stack and application related NV
variables.
InitAppConfig();
 // Initiates board setup process if button is
depressed
          // on startup
   if(BUTTON0_IO == 0)
   {
      SetConfig();
   }

StackInit();

#if defined(STACK_USE_HTTP_SERVER)
   HTTPInit();
#endif

#if    defined(STACK_USE_FTP_SERVER)    &&
defined(MPFS_USE_EEPROM)
   FTPInit();
#endif


#if          defined(STACK_USE_DHCP)          ||
defined(STACK_USE_IP_GLEANING)
   if(!AppConfig.Flags.bIsDHCPEnabled)
   {
      // Force IP address display update.
      myDHCPBindCount = 1;
#if defined(STACK_USE_DHCP)
      DHCPDisable();
#endif
   }
#endif


   // Once all items are initialized, go into infinite
loop and let
   // stack items execute their tasks.
   // If application needs to perform its own task, it
should be
   // done at the end of while loop.
   // Note that this is a "co-operative mult-tasking"
mechanism
   // where every task performs its tasks (whether
all in one shot
   // or part of it) and returns so that other tasks
can do their
   // job.
   // If a task needs very long time to do its job, it
must broken
   // down into smaller pieces so that other tasks
can have CPU time.

   while(1)
   {

          if(BUTTON0_IO == 0){
          while(BUTTON0_IO == 0){
                              }
          LCDErase();
          putrsUART("PR");
          getsUSART(cserial, sizeof(cserial));

          putrsUART("\r\nLa cadena es: ");
          putsUART(cserial);

          //StringToIPAddress(cserial, &tempIP)

          //strcpypgm2ram(LCDText,"Cadena: ");
```

```
            strcpy(LCDText,cserial);

            //strcpypgm2ram(LCDText, cserial);
            LCDUpdate();
}
// Blink SYSTEM LED every second.
if(TickGetDiff(TickGet(), t) >= TICK_SECOND/2 )
      {
         t = TickGet();
         LED0_IO ^= 1;
      }



if( TickGetDiff(TickGet(), t1) >= TICK_SECOND*10
)
      {
                              LCDErase();

                              putrsUART("PR");
                              getsUSART(cserial,
sizeof(cserial));


            //ReadStringUART(cserial,
sizeof(cserial));

            //cadserial=atoi(cserial[0]);
                              putrsUART("\r\nLa
cadena es: ");
                              putsUART(cserial);
                              strcpy(LCDText,cserial);
                              LCDUpdate();

                              t1 = TickGet();

StackTask();

#if defined(STACK_USE_HTTP_SERVER)
      // This is a TCP application.  It listens to TCP
port 80
      // with one or more sockets and responds to
remote requests.
      HTTPServer();
#endif

#if    defined(STACK_USE_FTP_SERVER)    &&
defined(MPFS_USE_EEPROM)
      FTPServer();
#endif

#if defined(STACK_USE_ANNOUNCE)
                        DiscoveryTask();
#endif

#if defined(STACK_USE_NBNS)
                        NBNSTask();
#endif

#if
defined(STACK_USE_GENERIC_TCP_EXAMPLE
)
                        GenericTCPClient();
#endif

      // In future, as new TCP/IP applications are
written, it
      // will be added here as new tasks.

      // Add your application speicifc tasks here.
      ProcessIO();


      // For DHCP information, display how many
times we have renewed the IP
      // configuration since last reset.
      if ( DHCPBindCount != myDHCPBindCount )
      {
         myDHCPBindCount = DHCPBindCount;

                              putrsUART(NewIP);
         DisplayIPValue(&AppConfig.MyIPAddr);
           // Print to UART
                              putrsUART(CRLF);
#if defined(STACK_USE_ANNOUNCE)
                              AnnounceIP();
#endif
      }

   }
}
```

La librería para el manejo del LCD queda de la siguiente manera:

```
//LCD
finctions//////////////////////////////////////////////////////////////////
//////////////////
void LCDdata (unsigned char value)
{
busylcd();
TRISE = 0;
DATALCD = value;
RS=1;
RW=0;
E=1;
LCDDelay();
E=0;
}
//-----------------------------------------------------------------------
--------------------
void LCDinit(void)
{
TRISE=0;
lcdcmd(0x38); //set 8-bit mode and 2 lines
lcdcmd(0x10); //cursor move & shift left
lcdcmd(0x06); //entry mode = increment
lcdcmd(0x0e); //display on - cursor blink on
lcdcmd(0x01); //clear display
}
```

```
//-----------------------------------------------------------------
------------------
void busylcd(void)
{
RS=0;
RW=1;
TRISE=255;
E=0;
LCDDelay();
E=1;
while(LCDBusy==1){
E=0;
LCDDelay();
E=1;
}
}
//-----------------------------------------------------------------
------------------
void lcdcmd(unsigned char temp)
{
busylcd();
RS=0;
RW=0;
TRISE=0;
DATALCD=temp;
E=1;
LCDDelay();
E=0;
}
//-----------------------------------------------------------------
------------------
void LCDDelay(void)
{
int i=0;
for (i=0;i<250;i++);

}
//-----------------------------------------------------------------
------------------
void stringtoLCD(unsigned char *m)
{
unsigned char i;
i = 0;
while(m[i] != 0)
{
LCDdata(m[i]);
i++;
}
}
void putrsXLCD(const rom char *buffer)
{
while(*buffer)                  // Write data to LCD up to
null       {
//while(BusyXLCD());      // Wait while LCD is busy
LCDdata(*buffer); // Write character to LCD
buffer++;               // Increment buffer
}
return;
}
void gotoxyXLCD(unsigned char x, unsigned char
y){
unsigned char direccion;
if(y != 1)
direccion = 0x40;
else
direccion=0;
direccion += x-1;
lcdcmd(0x80 | direccion);
}
```

La librería para la configuración del puerto USART queda de la siguiente manera:

```
#include "usart.h"
void putrsUSART(const rom char *data)
{
  do  {
  // Transmit a byte
    while(BusyUSART());
    putcUSART(*data);
  } while( *data++ );
}
void putcUSART(char data)
{
  TXREG = data;      // Write the data byte to the
USART
}
char BusyUSART(void)
{
            return !TXSTAbits.TRMT;
}
char getcUSART(void)
{
  return RCREG;                  // Return the
received data
{
  char i;    // Length counter
  unsigned char data;
  for(i=0;i<len;i++)  // Only retrieve len characters
            {
                        *Dest = '\0';
                        while(!DataRdyUSART());
                        c = getcUSART();
                        if(c == '\r' || c == '\n')
                                    break;
                        count++;
                        *Dest++ = c;
            }return count;
}
void putsUSART( char *data)
{
    do
```

```
  {  // Transmit a byte
    while(BusyUSART());                                    } while( *data++ );
    putcUSART(*data);

}
                        buffer++;
                        *buffer = '\0';
                        return;}
  }
}
buffer++;               // Increment the string pointer
            if(data=='\r'){
                        *buffer = '\n';
char DataRdyUSART(void)
{
            if(RCSTAbits.OERR) {
                        RCSTAbits.CREN = 0;
                        RCSTAbits.CREN = 1;
            }
  return PIR1bits.RCIF;
}
data = getcUSART();    // Get a character from the USART
  // and save in the string
    *buffer = data;

char ReadStringUSART(char *Dest,char

  {
    while(!DataRdyUSART());// Wait for data to be received

BufferLen)
{
            char c;
            char count = 0;
            while(BufferLen--)
}
void getsUSART(char *buffer, unsigned char len)
```

CÓDIGO DE APLICACIÓN WEB

```
<html>
<head>
        <title>Instituto Nacional de Ciencias
        M&#233;dicas y Nutrici&#243;n Salvador
        Zubir&#225;n</title>
        <link rel="stylesheet" type="text/css"
media="screen" href="estilo.css"/>
        <script type="text/javascript"
src="livevalidation_standalone.js"></script>
</head>

//Se ejecuta la función ajaxFunction() al acceder a
la página
<body onload="ajaxFunction()">

<script type="text/javascript">
   function ajaxFunction()
   {
      var xmlhttp;
/*
 Se crea el objeto XMLHttpRequest, que es un
elemento fundamental para la comunicación
asincrónica con el servidor.
La creación de este objeto varia si se trata de
Internet Explorer el cuál no incorpora JavaScript
sino que se trata de un ActiveX a diferencia de
otros navegadores como Firefox. Si el navegador
no soporta XMLHTTP manda una alerta al usuario
*/
      if (window.XMLHttpRequest)
      {
         //Código para: IE7+, Firefox, Chrome,
Opera, Safira…
         xmlhttp = new XMLHttpRequest();
      }

      else if (window.ActiveXObject) {
         xmlhttp = new
ActiveXObject("Microsoft.XMLHTTP");
      }
      else
      {
         alert("Tu buscador no soporta XMLHTTP");
      }
//Prepara la función de respuesta Se invoca cada
vez que se produce un cambio en el estado de la
petición HTTP.
      xmlhttp.onreadystatechange = function()
      {

      //readyState== 4 indica que se han recibido
todos los datos.
         if (xmlhttp.readyState == 4) {
                  var a1;
    //Se llama al contenido de la respuesta del
servidor en forma de cadena de texto separando
la información cada que encuentra una coma (,)
                  a1=xmlhttp.responseText;
                  a1=a1.split(',');
//Se le asigna el valor que se obtiene del servidor
a el que lleva por nombre param__ del formulario
dialisis
                   document.dialisis.param.value =
a1[0];
                  document.dialisis.param2.value
= a1[1];
                  document.dialisis.param3.value
= a1[2];
                  document.dialisis.param4.value
= a1[3];
                  document.dialisis.param5.value
= a1[4];
                  document.dialisis.param6.value
= a1[5];
                  document.dialisis.param7.value
= a1[6];
                  document.dialisis.param8.value
= a1[7];

                  //Código para obtener hora del
servidor
                  momentoActual = new Date()
                  hora =
momentoActual.getHours()
                  minuto =
momentoActual.getMinutes()
                  segundo =
momentoActual.getSeconds()
                  horaImprimible = hora + " : " +
minuto + " : " + segundo
                  document.dialisis.reloj.value =
horaImprimible

            setTimeout("ajaxFunction()",1000)

      }
 }
```

```
//Se prepara la aplicación para la respuesta del servidor. Se realiza una petición tipo GET que //no envía ningún parámetro al servidor y el archivo a abrir.

   xmlhttp.open("GET", "p1.php", true);
//Con el método send() se envian parámetros al servidor, si contiene null indica que no se envían parámetros al servidor.

   xmlhttp.send(null);
   }
</script>

/*Código página web
En este apartado se muestra parte de código de cómo se llama a la información en las distintas cajas de texto según corresponda*/
<form name="dialisis" method="post"
action="dialisis.php"  >
        <table style="width: 100%">
            <tr><td>
                Flujo dializante
                <input name="param5"
            type="text" size="8" style="width:
            71px" />
                <br />
            </td><td>
                Frecuencia cardiaca<input
                name="fc" type="text"
                size="8" style="width: 71px"
                />
                <br />
            </td></tr>
            <tr><td>
                Presi&#243;n de
                l&#237;nea arterial<input
                name="param2" type="text"
                size="8" style="width:
                71px"/><br />
                <br />

            </td><td>
                Frecuencia respiratoria
                    <input
                name="fresp" type="text"
                size="8" style="width:
                71px"/>
                <br />
            </td></tr>
            <tr><td>
                Presi&#243;n de
                l&#237;nea venosa<input
                name="param" type="text"
                size="8" style="width: 71px"
                />
                <br />
            </td></tr>

/* …  */
</form>
<body>
<html>
```

BIBLIOGRAFÍA

[1] Datos estadísticos INEGI. "Porcentaje de casos de morbilidad hospitalaria por entidad federativa y principales causas según sexo, 2001 a 2006"
URL http://www.inegi.org.mx/est/contenidos/espanol/rutinas/ept.asp?t=msal05&s=est&c=3356
Fecha de Consulta Enero 2010

[2] "Operators Manual FDS-2008H", Fresenius, USA 1988.

[3] Berrocal V, Gil S, Ossa E, Romero A, "Sistema de Telemonitoreo de Signos Vitales utilizando una Red LAN", Universidad Autónoma del Caribe, Cuba 2008.

[4] "Diseño y Programación de una base de datos con parámetros fisiológicos para una central de máquinas de Hemodiálisis Fresenius 2008H",
Tesis de Ingeniero Biomédico, Lara Navarrete Samuel, Unidad Profesional Interdisciplinaria de Biotecnología del Instituto Politécnico Nacional, México 2007

[5] Olmos k, Gómez M, Rascón L, "Desarrollo de una interfaz de Usuario para Máquina de Hemodiálisis", Universidad Autónoma de Ciudad Juárez, México 2007.

[6] Vender, Sherman, Luciano, "Fisiología Humana", Editorial McGRAW-HILL, Segunda Edición, México 1978

[7] Guyton,Hall, "Fisiología Humana", Editorial McGRAW-HILL, Decima Edición, México 2001.

[8] "Tema 10. Insuficiencia Renal", Facultad de Estudios Superiores Zaragoza, Universidad Nacional Autónoma de México",
URL http://www.zaragoza.unam.mx/educacion_n_linea/tema_10_insuf_renal/t10antecedentes.html
Fecha de Consulta Agosto de 2009

[9] Traducción. Dr Tango. Inc. Versión en inglés PATEL, Parul, "Review provided by VeriMed Halthcare Network and David Zieve Medicla Director. A.D.A.M. Inc".
URL http://www.nlm.nih.gov/medlineplus/spanish/ency/article/000501.htm
Fecha de Consulta Agosto 2009

[10] Equipo editorial de Fisterra, Médicos especialistas en Medicina de Familia y en Medicina Preventiva y Salud pública, "Hemodiálisis. Información para pacientes sobre hemodiálisis",
URL http://www.fisterra.com/salud/3proceDT/hemodialisis.asp
Fecha de Consulta Agosto de 2009

[11] Baxter "Terapias. ¿Qué es la Hemodiálisis?",
URL http://www.latinoamerica.baxter.com/argentina/terapias/sub/terapias_ter_renal_hd.html
Fecha de Consulta Agosto de 2009

[12] Medimexico, "Información de Dialisis",
URL http://www.medimexico.com.mx/info_e/info_e.html#top
Fecha de Consulta Agosto de 2009

[13] AAKP American Association of Kidney Patients (Asociación Americana de Pacientes Renales), Conozca sus Opciones de Hemodiálisis. Vol. I USA 2009

[14] Carrera Ingeniería Electrónica en la Universidad Antonio Nariño en Colombia, Sur América, "Manual de Microcontroladores Microchip"
URL http://www.uniovi.es/ate/alberto/manualPic.pdf
Fecha de Consulta Agosto 2009

[15] Facultad de Informática de la Universidad Complutense de Madrid “Programación en ensamblador”
URL http://www.fdi.ucm.es/profesor/mchagoye/lfc0708/lfc_parte2.pdf
Fecha de Consulta Agosto 2009

[16] García Breijo, “Compilador C CCS y Simulador PROTEUS para Microcontroladores PIC” Editorial Alfaomega, Primera Edición México 2008.

[17] Herrero G. “Lenguaje de Programación C18. Introducción”
URL http://slalen.ifastnet.com/pics/datos/C18.pdf
Fecha de Consulta Octubre 2009

[18] Muhammad Ali Mazidi, “The 80x86 IBM PC and Compatible Computers Vol II2, Pc Interno, USA 1999

[19] De Rueda Luis Tutorial puerto paralelo,
URL http://perso.wanadoo.es/luis_ju/soft/files/puerto.pdf
Fecha de Consulta Octubre 2009

[20] Axelson Jan, “Everything you need to develop custom USB peripherals”, Editorial Madison Lakeview Research LLC Tercera Edición, USA 2005,

[21] Universidad Nacional del Nordeste, Facultad de Ciencias Exactas Naturales y Agrimensura Cátedra:Teleproceso y Sistemas Distribuidos, Profesor La Red Martínez, David Luis.
http://exa.unne.edu.ar/depar/areas/informatica/SistemasOperativos/TECNOLOGIASETHERNET,FRAMERELAY.pdf
Fecha de Consulta Agosto del 2009

[22] Universidad de Colima, Facultad de Telemática, Materia Redes Locales, Ing. Juan Antonio Guerrero
URL http://docente.ucol.mx/al985660/public_html/token%20ring.html
Fecha de Consulta Agosto del 2009

[23] Ponce, Tortosa, Maicas “Manual de Redes Inalámbricas IEEE 802.11”,
http://www.wi-fi.com/3450054
Fecha de Consulta Agosto del 2009

[24] Universidad de las Américas Puebla, Escuela de Ingenieria y Ciencias,Jibran De La Rosa Ramos
“Seguridad de Redes inalámbricas IEEE 802.11 (WLAN)”, Capitulo2,
URL http://catarina.udlap.mx/u_dl_a/tales/documentos/lem/de_l_j/capitulo2.pdf
Fecha de Consulta Octubre 2009

[25] National Instruments. “Punto de Inicio en Comunicación Serial.”
URL http://www.mundopc.net/hardware/componen/puertos/
Fecha de Consulta Octubre 2009

[26] Granada Carrillo Marlon, Madrid España “Brazo Robotico-MAX232,”
URL:http://www.madridbot.org/Documentos/Documentos%202009/Prueba%20Libre/Memoria%20Brazo%20Robotico.pdf
Fecha de Consulta Octubre 2009

[27] Escuela Politécnica superior de Valencia “Diseño y Programación de un DSPIC para conexión Ethernet”, Laboratorio de Sistemas Electrónicos Digitales, España 2008

[28] Stallings William, “Comunicaciones y Redes de Computadores”, Editorial Pearson Prentice Hall, Cuarta Edición, España 2004.

[29] Tanenbaum Andrew S., "Redes de Ordenadores", Editorial Prentice Hall Hispanoamericana, Quinta Edición, España 2005

[30] Diego López García "Fundamentos de comunicaciones y redes de datos", Escuela Politécnica Superior, Universidad de Huelva, España 2006.

[31] Advantage Database Server "Cliente/Servidor para archivos DBF"
URL http://www.ciber-tec.com/ads.htm
Fecha de Consulta Agosto de 2009

[32] DUQUE, Méndez Néxtor Darío "Bases de Datos", Universidad Nacional de Colombia, Facultad de Administración, Departamento de Informática y Computación,
URL http://www.virtual.unal.edu.co/cursos/sedes/manizales/4060029/lecciones/cap6-1.html
Fecha de Consulta Agosto de 2009

[33] Deep in PHP, "Modelo Vista Controlador (MVC),
URL http://www.deepinphp.com/category/arquitectura/
Fecha de Consulta Agosto de 2009

[34] VARGAS. R, MALTES. Juan, "Programación en capas", Universidad de Costa Rica, Ciencias de Computación e Informática.
URL http://www.di-mare.com/adolfo/cursos/2007-2/pp-3capas.pdf
Fecha de Consulta Agosto de 2009

[35] Editum.org "¿Qué es un servidor de aplicaciones?"
URL http://www.editum.org/Que-Es-Un-Servidor-De-Aplicaciones-p-473.html
Fecha de Consulta Agosto de 2009

[36] "Servidor de Aplicaciones"
http://wiwiloz.wordpress.com/servidor-de-aplicaciones/
Fecha de Consulta Agosto de 2009

[37] Diccionario Informático, "Definición de Servidor de aplicaciones",
URL http://www.alegsa.com.ar/Dic/servidor%20de%20aplicaciones.php
Fecha de Consulta Agosto de 2009

[38] VOLODARSKY, Mike, "Las diez principales mejoras de rendimiento en IIS 7.0", TechNetMagazine.
URL http://technet.microsoft.com/es-es/magazine/2008.09.iis.aspx
Fecha de Consulta Agosto de 2009

[39] "Definiciòn de Apache", Master Magazine
URL http://www.mastermagazine.info/termino/3866.php
Fecha de Consulta Agosto de 2009

[40] FLETESn HERNÁNDEZ, REYES, "Arquitectura de base de datos", Instituto Tecnológico de Colima, Departamento de Sistemas y Computación, Tutorial de fundamentos de bases de datos
URL http://labredes.itcolima.edu.mx/fundamentosbd/sd_u1_4.htm
Fecha de Consulta Agosto de 2009

[41] SILBERSCHATZ, KORTH, SUDARSHAN, "Fundamentos de Bases de Datos", 5ta edición, España 2006, Editorial Mc Graw Hill.

[42] CAMPOY, Lourdes, "Tutorial de Base de Datos I", Departamento de sistemas y computación, Tecnológico La paz,
URL http://sistemas.itlp.edu.mx/tutoriales/basedat1/portada.htm

Fecha de Consulta Agosto de 2009
[42] CAMPOY, Lourdes, “Modelo entidad-relación”, Departamento de sistemas y computación, Tecnológico La paz,
URL http://sistemas.itlp.edu.mx/tutoriales/basedat1/tema4_2.htm
Fecha de Consulta Agosto de 2009

[43] POZO, Salvador, “MySQL con Clase”, Marzo de 2004,
URL http://mysql.conclase.net/curso/index.php?cap=004
Fecha de Consulta Agosto de 2009

[44] Con clase.net, “ Dependencia funcional, primera y segunda forma normal”.
URL http://mysql.conclase.net/curso/index.php?cap=004a#NOR_dependenciafuncional
Fecha de Consulta Agosto de 2009

[45] WHITE, Phil, “The real story of Informix software”, Sand Hill Publishing, California.
URL http://www.storyofinformix.com/TheRealStoryofInformixandPhilWhite.pdf
Fecha de Consulta Agosto de 2009

[46] “Informix Dynamic Server 11.5. Features and benefits”, IBM,
URL http://www-01.ibm.com/software/data/informix/ids/feature.html
Fecha de Consulta Agosto de 2009

[47]”Oracle’s History: Innovation, leadership, results”, ORACLE Corporation,
URL http://www.oracle.com/corporate/story.html
Fecha de Consulta Agosto de 2009

[48] “Sobre PostgreSQL”, Portal en español sobre PostgreSQL,
URL http://www.postgresql-es.org/principal
Fecha de Consulta Agosto de 2009

[49]”Company Background” dBase,
URL http://www.dbase.com/About_us.asp
Fecha de Consulta Agosto de 2009

[50] “MySQL 5.0 Manual de Referencia.”, MySQL & Sun Microsystems,
URL http://dev.mysql.com/doc/refman/5.0/es/introduction.html
Fecha de Consulta Agosto de 2009

[51] Full Web Building Tutorials,
URL http://www.w3schools.com/default.asp
Fecha de Consulta Agosto de 2009

[52] Java Sun
URL http://java.sun.com/products/jsp/
Fecha de Consulta Agosto de 2009

[53] ALVAREZ, Miguel, “La tecnología Java para la creación de páginas web con programación en el servidor”, Desarrollo web
URL http://www.desarrolloweb.com/articulos/831.php
Fecha de Consulta Agosto de 2009

[54] EGUÍLUZ. Javier, ”Introducción a AJAX”, Creative Commons.
URL http://www.librosweb.es/ajax
Fecha de Consulta Agosto de 2009

[55] Tarjeta de Desarrollo PICDEM.net 2 Internet/Ethernet. Guía de Usuario, Microchip 2006

[56] Burhart, M. “Introducción a los servidores Web Dedicados”.
URL:http://www.daleya.com/?query=PILA+TCP%2FIP+MICROCHIP&chat=activo&daleyaidext%5B%5D=1&ext
Fecha de Consulta Agosto de 2009

[57] Herramienta de Desarrollo MPLAB Versión 8.3 , Microchip Corporation ,
URL:http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002#P140_5618,
Fecha de consulta Agosto 2009

[58] Programador PICKit II, Microchip Corporation,
URL: http://www.microchipdirect.com/productsearch.aspx?Keywords=DV164120,
Fecha de consulta Enero 2010

[59] “The 8 most successful open source products ever”
URL http://royal.pingdom.com/2009/05/29/the-8-most-successful-open-source-products-ever/
Fecha de Consulta Agosto de 2009

[60] “Internet 2008 in numbers”
http://royal.pingdom.com/2009/01/22/internet-2008-in-numbers/
Fecha de Consulta Agosto de 2009

[61] Brey Barry, Microprocesadores Intel, Editorial Prentice Hall, 7ª Edición, México 2006

[62] Compilador C18 Version Educativa, Microchip Corporation,
URL:http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en536656
Fecha de consulta Enero 2010

RESUMEN

El presente proyecto consiste en implementar un sistema de monitoreo para máquinas de Hemodiálisis Fresenius basado en la utilización de un microcontrolador PIC18F97J60 y un controlador Ethernet ENC28J60. El sistema está conformado por dos partes principales: transmisión y monitorización de datos.

El sistema de transmisión de datos que consiste en enviar los parámetros de la máquina de Hemodiálisis vía interfaz serial, y al mismo tiempo dichos parámetros son procesados e interpretados por el microcontrolador el cual tiene funcionalidad HTML, generándose una página web donde se ven reflejados.

La segunda parte, consiste en un formulario que manda a llamar los parámetros almacenados en la página web guardada en el microcontrolador, mostrarlos y posteriormente almacenarlos en una Base de Datos.

ABSTRACT

The present project consists of the implementation of Hemodyalisis monitoring system to Fresenius Machines which is based on the use of one PIC18F97J60 microcontroller and one Ethernet controller. The system is composed by two main parts: data transmission and data monitoring.

The data transmission lies in sending the parameters of Hemodyalisis' machine via serial interface at the same time the parameters are treated and interpreted by the microcontroller, which has HTML functionality, and are reflected on a web page.

The second part consists of a form which calls the parameters stored on the microcontroller's web page, displays them and subsequently the information is stored in a database.