UNIVERSIDAD TECNOLOGICA NACIONAL
FACULTAD REGIONAL SAN NICOLAS

INGENIERIA EN ELECTRONICA

PROBLEMA DE INGENIERÍA

TECNICAS DIGITALES III

CENTRAL DE MONITOREO PARA MONITORES MULTIPARAMÉTRICOS

Integrantes**:**

- Caporalini, Leonardo D.
- Morales, Matías G.

Docentes**:**

- Profesor: Poblete Felipe
- Auxiliar: Gonzalez Mariano

AÑO 2011

INDICE

OBJETIVOS DEL TRABAJO

Este trabajo tiene por objetivo diseñar e implementar una central de monitoreo para monitores multiparamétrico marca Feas en una Unidad de Terapia Intensiva.
La necesidad de realizar este proyecto se debe a que existen dos camas que se encuentran alejadas del sector donde están los enfermeros del servicio, no pudiendo desde la posición en la que están ver los parámetros vitales de los pacientes.
Ante la imposibilidad de compra por parte del hospital de una central de monitores a la empresa que la produce (por cuestiones burocráticas del propio hospital) y debido a que el equipo de Electromedicina, al cual pertenecemos, cuenta con las herramientas necesarias para el desarrollo, se evalúa la posibilidad de realización.
Luego de algunas consultas con profesionales del lugar, respecto de las necesidades que debe cubrir la central, y con Ingenieros del hospital especializados en normalización de equipos, se le ofrece al servicio realizar un equipo, con limitaciones en cuanto funciones respecto a la central original, pero cubriendo gran parte de la necesidad de ese sector.
Cabe aclarar, que este proyecto no reemplaza a una central original propietaria de los monitores sino que trata de realizar algunas de sus funciones a fin de cumplimentar el funcionamiento de los equipos a los cuales está conectado. Las limitaciones de este proyecto respecto a una central se detallaran a lo largo de este informe.
También remarcamos que tanto este equipo como los monitores a los cuales están conectados no son de sostén de vida, sino que sirven para realizar monitoreo de los parámetros del paciente.

MATERIAS INTEGRADAS

• Electrónica Aplicada II.
Se aplicaron conocimientos en amplificadores operacionales.

• Técnicas Digitales II
Manejo y programación de microcontroladores PIC.

• Medidas Electrónicas II
Conversión A/D, cuantificación, mediciones de señales con ruido.

POSIBLES APLICACIONES

• Este proyecto fue pensado para este caso en particular, pero con algunas modificaciones puede ser empleado para tomar otros parámetros o bien distinto número de camas.
La ventaja de ser un tendido cableado desde el equipo hacia la PC es que esto puede ser utilizado en servicios donde tiene paredes plomadas debido al uso de equipos de Rx y al contar con una fuente con un filtro FI, el rechazo al ruido de línea es excelente.

BIBLIOGRAFÍA

•*Amplificadores operacionales y circuitos integrados lineales 5ta edición – Coughlin-Driscoll*

•*Teoría y diseños con microcontroladores PIC- Antonio Tafanera*

•*Ayuda MSDN de Visual Basic 6.0*

•*Hoja de datos de mcHID.dll*

•*Apuntes de Cátedra de Medidas Electrónicas II y Electrónica Aplicada II*

•*Manual técnico de monitores Feas mod. Multipar LCD*

•*Normas de Compatibilidad Electromagnética*
•*Datasheets de: PIC18F2550*
CA3140A

DESARROLLO

INTRODUCCIÓN

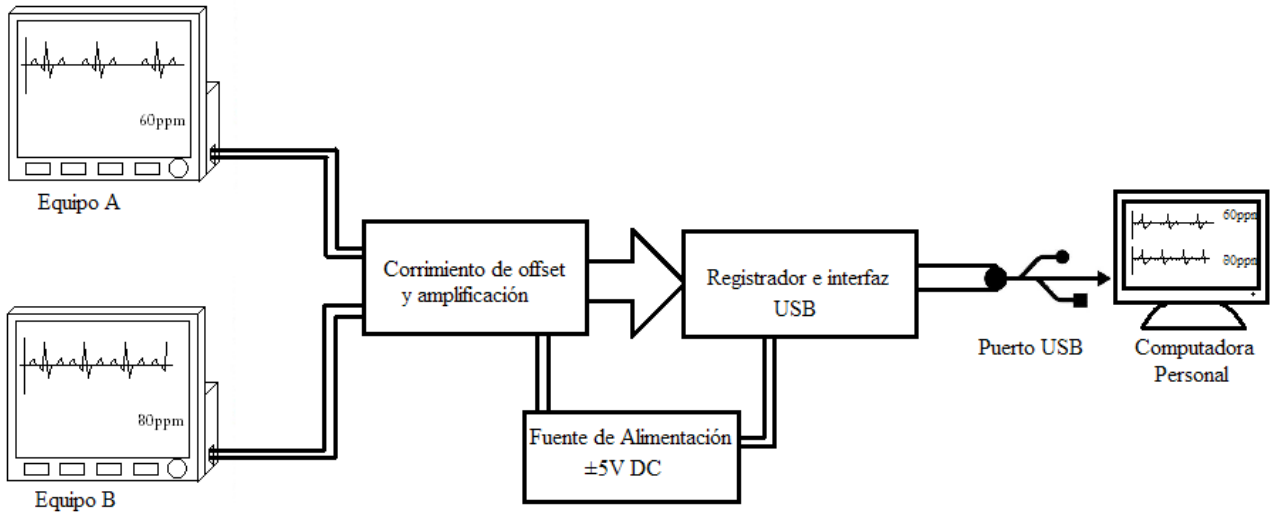

Fig 1 – Diagrama de bloques del sistema

Como podemos ver en el esquema, este proyecto cuenta con unidades fácilmente distinguibles; por un lado, la adecuación de la señal de entrada en cuanto a niveles de tensión y offset, la conversión A/D y registro y, por otro lado, una vez digitalizada en la PC la representación en pantalla de la señal y su análisis en base a los datos que entregan los equipo. Por último, también tenemos una fuente de alimentación partida para alimentar tanto a los amplificadores operacionales como al microcontrolador.
Antes de continuar describiendo el funcionamiento, vamos a definir algunos términos que empleamos en este proyecto:

MONITOR MULTIPARAMÉTRICO
Un monitor multiparamétrico es un equipo destinado a la visualización y adquisición simultánea de varios parámetros biológicos, como pueden ser ECG, presión sanguínea, oxigeno en sangre, capnografía, etc. Existen en el mercado diversos modelos con múltiples grados de complejidad, exactitud, cantidad de señales y posibilidad de conexión a central de monitoreo.
En nuestro caso particular, lo monitores utilizados cuentan con una ficha de conexión en la cual envía de manera analógica las señales principales hacia una central por medio de una ficha DB9 ubicada en el lateral derecho del equipo. La imagen de equipo y su conexión con central de muestran en las figuras 2 y 3.

Fig 2 – Monitor multiparamétrico

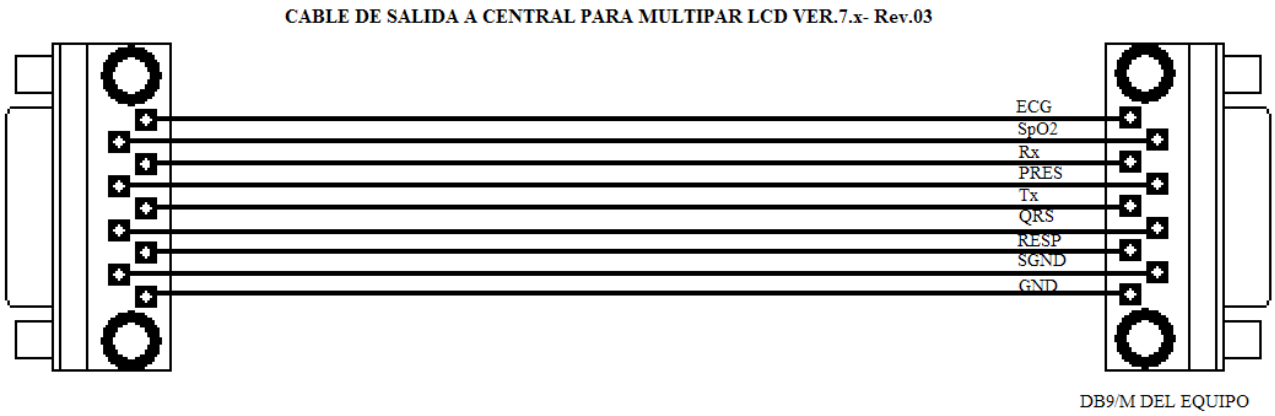

Fig 3 – Cable de conexión del monitor con la central

CENTRAL DE MONITOREO

Una central de monitoreo es un equipo en el cual se pueden visualizar los parámetros biológicos de pacientes que se encuentran a distancia o su monitor se encuentra posición en la cual no es posible visualizase desde el puesto en donde están los médicos o enfermeros.

En general, las centrales de monitoreo son para una marca específica de monitores ya que no existe en la actualidad un estándar de comunicación entre centrales y equipos. En nuestro caso, nuestro proyecto se centrará en realizar algunas de las funciones de una central de monitoreo marca Feas, para monitores Multipar LCD de la misma marca.

Fig 4 – Central de monitoreo

TEORÍA DE FUNCIONAMIENTO

En la introducción se adelanto parte de cómo es el funcionamiento del proyecto, en esta sección vamos a describir en mayor profundidad el funcionamiento de cada uno de los bloque que describen el sistema por completo.

Vamos a comenzar por analizar las señales de entrada a nuestro equipo. Estas son señales analógicas que tienen la misma forma que la curva que se representa en pantalla. Además, los niveles de tensión de la señal de ECG de salida a central no son compatibles con los niveles de tensión de entrada que maneja el microcontrolador, es por esto que además de digitalizar la señal, debemos acondicionarla de manera tal que además de entrar en el rango, se aproveche una mayor excursión de señal y por lo tanto una mejor calidad de la señal muestreada.

Para realizar el circuito de acondicionamiento, se utilizó un sumador inversor por canal de entrada de este tipo de señal (en este proyecto necesitaremos dos). Su diagrama y principio detallado de funcionamiento será explicado en el apartado siguiente, por tanto ahora digamos que esta sección amplifica la señal y le suma una tensión de offset para aprovechar al máximo la excusión de señal

de los conversores A/D del PIC cuyo rango es de 0 a 5 V. De todos modos se dejó una tensión de guarda por si la señal varía en su amplitud y como para también no saturar las entradas.
Una vez acondicionada, la señal es muestreada y convertida de analógica a digital por el PIC. El microcontrolador que vamos a utilizar para este proyecto es el PIC18F2550 de Microchip que cuenta con interfaz USB y cumple con todos nuestros requerimientos.
Los datos que obtuvimos de la hoja de datos fueron los siguientes:

MÓDULO CONVERTIDOR DE 10-BIT ANALÓGICO A DIGITAL (A/D)
El módulo conversor de analógico a digital (A/D) tiene 10 entradas con los dispositivos de 28 pines y 13 en los de 40/44 pines. Este módulo permite la conversión de una señal de entrada analógica a un número digital de 10 bits.
El módulo tiene cinco registros:
• Registro alto del resultado A/D (ADRESH)
• Registro bajo del resultado A/D (ADRESL)
• Registro de control A/D 0 (ADCON0)
• Registro de control A/D 1 (ADCON1)
• Registro de control A/D 2 (ADCON2)
El registro ADCON0, gobierna las operaciones del módulo A/D. El registro ADCON1 configura las funciones de los puertos. El registro ADCON2 configura el reloj, programa el tiempo de adquisición y justificación.

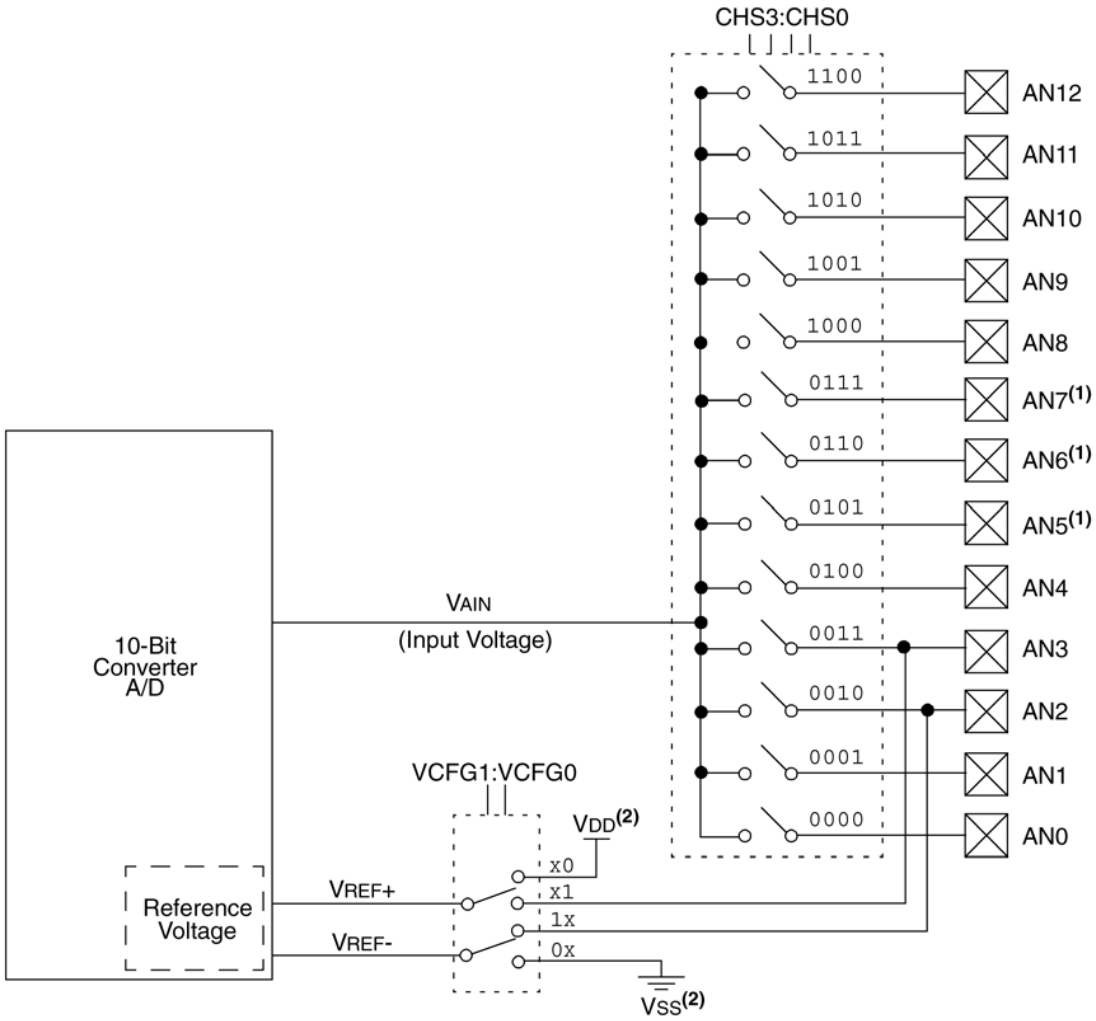

Fig5 – Diagrama de bloques del A/D

(1): Los canales de 5 a 7 no están disponibles en los dispositivos de 28 pines.
(2): Los pines de E/S tienen un diodo de protección a VDD y VSS.

Después de que el módulo A/D se haya configurado según lo deseado, el canal seleccionado debe adquirir antes de que comience la conversión.
Los canales de entrada analógica deben tener sus bits TRIS correspondientes seleccionados como entrada.
Después de esta adquisición el tiempo ha transcurrido, la conversión A/D puede comenzar.
Se puede programar un periodo de adquisición para que ocurra entre activar el bit GO/DONE* y el comienzo real de la conversión

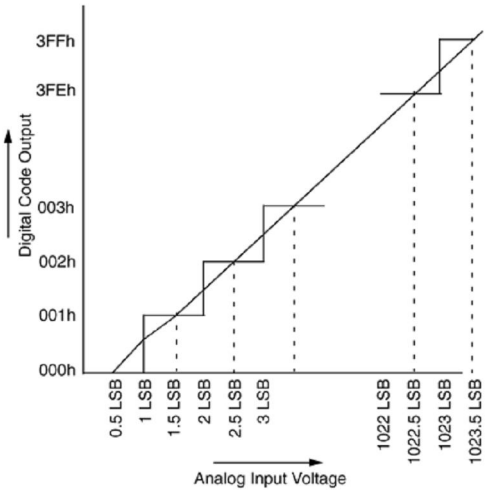

Fig6 – Función de transferencia A/D

El encargado de realizar el registro y de realizar una interfaz USB será el programa que cargaremos en el PIC. Este está programado en lenguaje Basic y para realizarlo y compilarlo usamos el MicroCode Studio ya que este tiene una gran cantidad de librerías que si son usadas de manera correcta nos ahorra mucho código y tiempo si comparamos con la programación en assembly.
Como primera medida el programa inicializa y define los parámetros de conversión e interfaz USB y variables a utilizar las cuales son buffers de USB y variables en las que serán almacenadas las conversiones del A/D. Además, configura cuales serán los puertos y pines de entradas analógicas.
Una vez inicializados estos parámetros, comienza el programa principal el cual usa las rutinas de adquisición de la señal analógica. Una vez adquirido el dato, se llama a un subprocedimiento en el cual se establece la conexión y entrega de los datos adquiridos por medio de librerías destinadas al manejo del puerto USB. Ya realizado el envío de los datos, vuelve al programa principal para adquirir nuevamente datos de la señal de entrada, iniciando nuevamente el ciclo.
Para entender cómo funciona el módulo USB del microcontrolador nos basamos en su hoja de datos. Los datos que obtuvimos fueron:

BUS SERIE UNIVERSAL (USB)
DESCRIPCIÓN DEL PERIFÉRICO USB
La familia de dispositivos PIC18FX455/X550 contiene una interfaz serie compatible con el SIE (serial interface engine, máquina con comunicación serie) USB “full-speed” (2.0) y “de poca velocidad” (1.0) que permite la comunicación rápida entre cualquier dispositivo USB y el microcontrolador PIC®.
El SIE puede interconectarse directamente al USB, utilizando el transreceptor interno, o puede conectarse a través un transmisor-receptor externo. El PIC tiene un regulador interno de 3,3V para accionar el transmisor-receptor interno en aplicaciones de 5V.
Se han incluido algunas características especiales en el hardware para mejorar el funcionamiento. Se proporciona memoria de puerto dual en la memoria de datos del dispositivo (RAM del USB) para tener acceso directo a la memoria desde el núcleo del microcontrolador y desde el SIE. También se proporcionan unos buffer para que el programador elija libremente el final de la memoria dentro del espacio de la RAM del USB. Existe un puerto paralelo para transmitir datos grandes, por ejemplo datos al puerto paralelo, se ha proporcionado la ayuda de transferencia

ininterrumpida de volúmenes de datos grandes, por ejemplo datos síncronos, a los buffer de memoria externos.
El diagrama de bloques de los periféricos y opciones del USB es el siguiente.

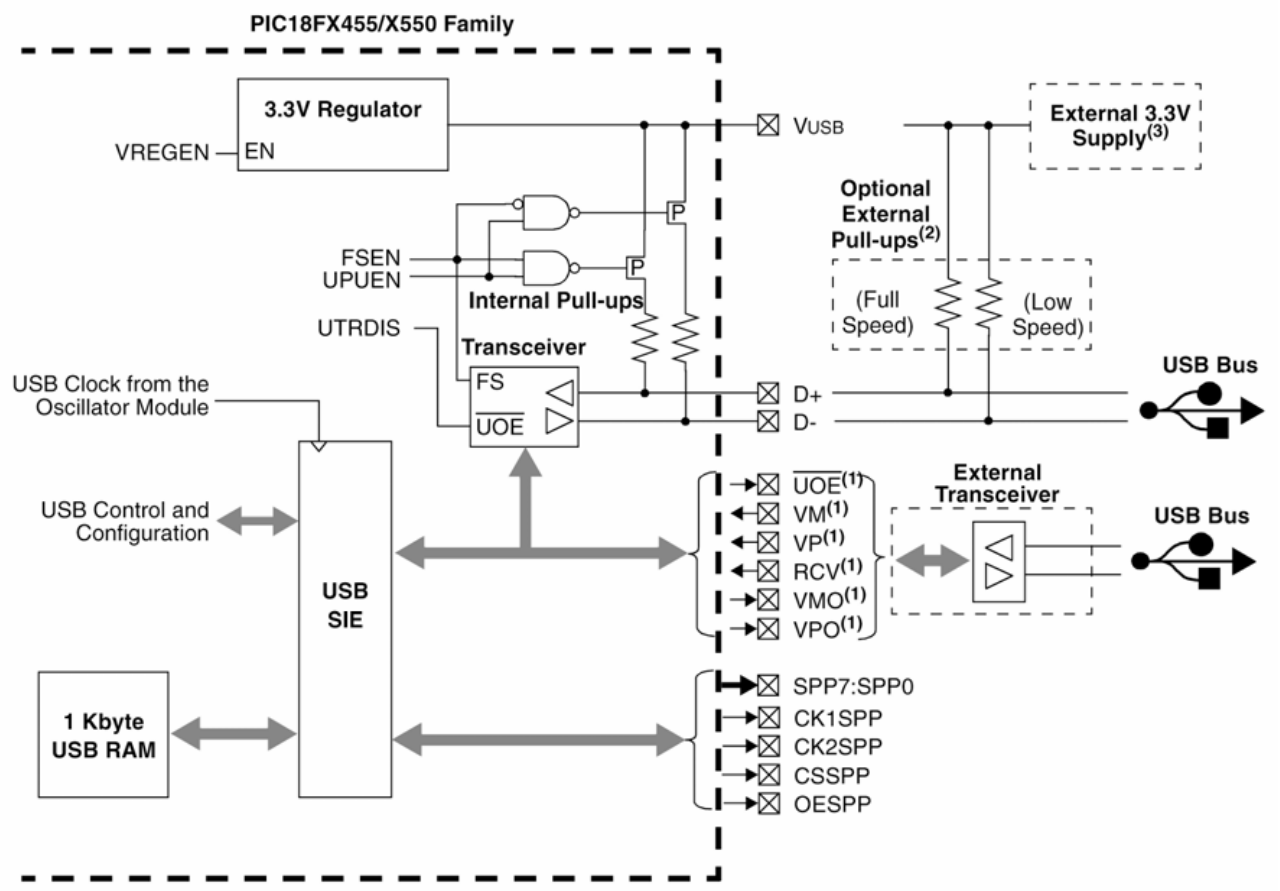

Fig. 7 – Periférico y opciones del USB

(1) Esta señal solo está disponible si el transmisor interno está desactivado (UTRDIS=1).
(2) Las resistencias internas pull-up se tienen que desactivar (UPUEN=0).
(3) No hay que activar el regulador interno cuando se utiliza una fuente de 3,3 V externa.

Una vez que tenemos el dispositivo enviando datos a la PC, desde aquí debemos levantar los datos desde el puerto para mostrarlo en pantalla. Para realizar esto, de realizó un programa en Microsoft Visual Basic 6.0 que toma las conversiones realizadas por el PIC y no solo las grafica, sino que también infiere el numero de pulsaciones por minuto (ppm) en base a la curva entre otras funcionalidades.
El código de programa con sus comentarios y capturas de pantallas, se encuentran en las secciones siguientes.
De manera general, el programa levanta la información enviada por el PIC, adecúa ese valor numérico al Picture Box en base al zoom, posición y offset automático y realiza una línea desde el valor anterior al actual.
En paralelo a la rutina de graficar la señal, se realiza un cálculo de ppm. Para hacerlo se tomaron como base los apuntes de Medidas Electrónicas II con respecto a la medición de período.
Para contar las ppm generamos una base de tiempo fija, un valor de referencia variable y creamos una bandera; entonces para sacar el período entre una pulsación y otra hacemos así: cuando la señal supera el valor de referencia, la bandera se coloca en 1. A partir de ahí contamos timers y los almacenamos en un contador, cuando la señal es menor que la referencia colocamos la bandera

nuevamente en 0 y seguimos contando hasta que la señal de ECG vuelve a sobrepasarlo. En ese instante obtenemos las ppm como:

ppm = Int(((nº de veces que el timer entra en un segundo / (contador)) * 60))

Una vez realizada la cuenta, se muestra por pantalla el valor calculado.
El programa para un mejor cálculo de las ppm y muestreo por pantalla incorpora un algoritmo de corrimiento y actualización de offset automático en función del valor de la componente de continua que presente la señal.
Además, se incorporaron señalizaciones de conexión y habilitación del monitor y hardware desarrollado y seteo de alarmas como se verá en el código del programa.

El Sistema Operativo elegido para la PC, es Windows Xp, ya que es uno de los SO mas extendido, y además se le permitirá a los usuarios utilizar el paquete de Office para algún eventual informe.
Se eligió Windows XP SP3 que es la versión más actual de este sistema operativo, lo que habrá que definir el costo de la licencia, a la fecha actual.
Como medida ante posibles infecciones, virus o una mala utilización de la computadora, utilizaremos políticas de grupo para usuarios. Se limitara el acceso al usuario Administrador con contraseña, y se creara un usuario exclusivo, sin contraseña, para su utilización.
Desde la cuenta de administrador, se utilizara el software, gratuito Windows SteadyState cuya versión actual es la 2.5. Este software permite administrar los usuarios de Windows XP (crear, importar, exportar a otra computadora). Creando perfiles de usuarios, se limitara en cada perfil el uso del menú inicio, evitar que abran el panel de control, editor del registro, administrador de tareas, se bloquearan funciones de internet explorer e incluso el acceso a determinados programas y carpetas.

Se le limitará todo lo referente al uso inadecuado, permitiendo solo el acceso al software creado y al paquete de Office.
También para prevenir infecciones de virus y el mal uso se limitara la utilización de dispositivos extraíbles (pendrives, cámaras fotográficas, celulares, etc.), y no se proveerá con lectora de CD.

Ante un eventual reinicio del CPU, se agrego una porción de código, para que el programa inicie automáticamente con Windows. Para ello se investigo cuál es la parte del registro de Windows que permite esta operación y los procedimientos en Visual Basic para escribir sobre el registro.
Primero se debe crear una clave en la rama del registro

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\

El programa una vez iniciado (por primera vez) primero se fija si no está creada la clave, de no estarlo la crea, para asi iniciar automáticamente al iniciar Windows.
Esto se consigue utilizando una clase llamada cQueryReg. Basándonos en su documentación, realizamos el siguiente codigo.

```
Private mReg As cQueryReg
Private Const cvRun As String =
“HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\"

'En el procedimiento de carga de la aplicación, vefiricamos/creamos la clave en
el registro a
‘traves de la clase creada

Set mReg = New cQueryReg
Dim s As String
' Poner la clave
Dim s As String
'
s = mReg.GetRegString(cvRun, App.EXEName)
If s = "" Then
    If mReg.SetReg(cvRun, App.EXEName, App.path +”\” +App.EXEName+”.exe” ) <>
ERROR_NONE Then
        LabelInfo.Caption = "ERROR al crear la clave de Inicio"
    End If
End If
```

CIRCUITOS Y DIAGRAMAS

El circuito consta de tres placas, una es en la cual se encuentra el PIC y está formada por el cable de conexión USB, borneras de entrada de tensión, y componentes pasivos para que el micro funcione. Otra sección es la que realiza un acondicionamiento de las señales de entrada y por último, la fuente de alimentación la cual cuanta en su entrada con un filtro.

PLACA DE REGISTRO E INTERFAZ USB

En esta placa se encuentra el corazón del proyecto el cual es el PIC18F2550 de Microchip, y otros componentes que hacen posible su funcionamiento. Los pines de entradas analógicas se encuentran disponibles por medio de borneras así como también bornes de gnd para las señales y bornes de conexión USB. También se agregaron del LEDs para mostrar si el monitor esta conectado como se puede ver en las figuras 8a, 8b y 9.

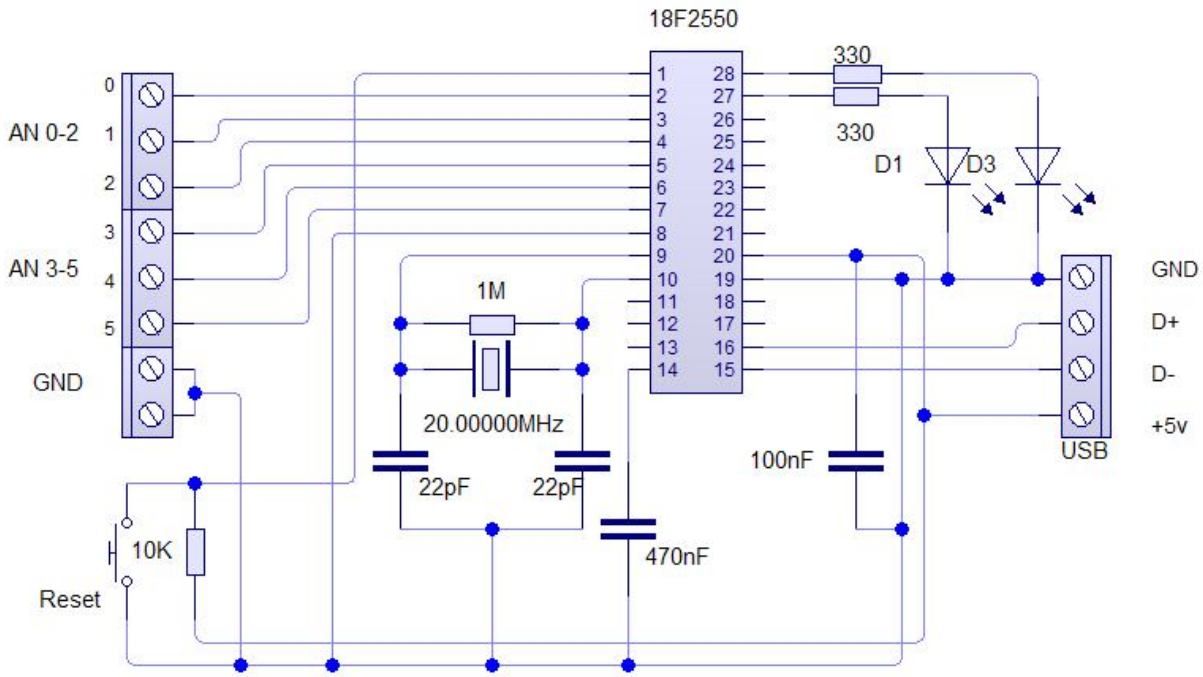

Fig8a – Diagrama de la placa de registro e interfaz USB

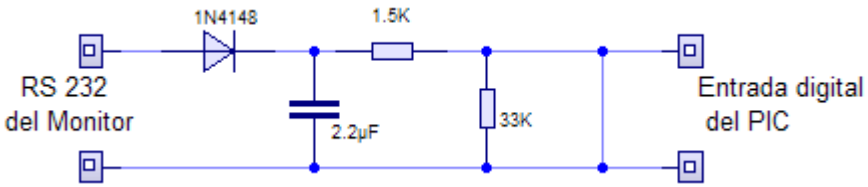

Fig8b – Diagrama de ficha para la detección de monitor conectado

En la figura anterior, se muestra como se toma la señal de conexión detectando el pico de rs232, el cual se descarga y reduce a través de las resistencias.

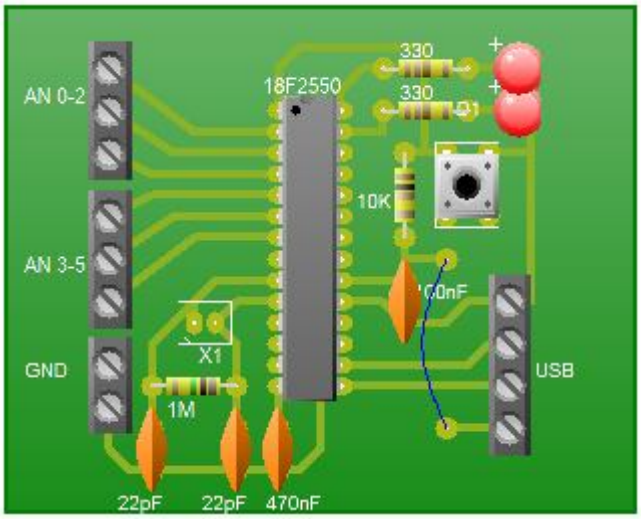

Fig9 – PCB de la placa de registro e interfaz USB

PLACA DE AMPLIFICACIÓN Y CORRIMIENTO DE OFFSET

Este circuito es básicamente un amplificador sumador inversor, donde la señal de ECG de entrada se amplifica y se le suma una tensión continua de offset debido a que originalmente tiene un valor menor a cero en parte de la señal no pudiendo el PIC tomarla. En la figura 10 podemos ver la señal de entrada y de salida simulada. De esta manera se observa cómo se aprovecha la excursión de señal a la entrada del PIC, la cual es de 0 a 5v.

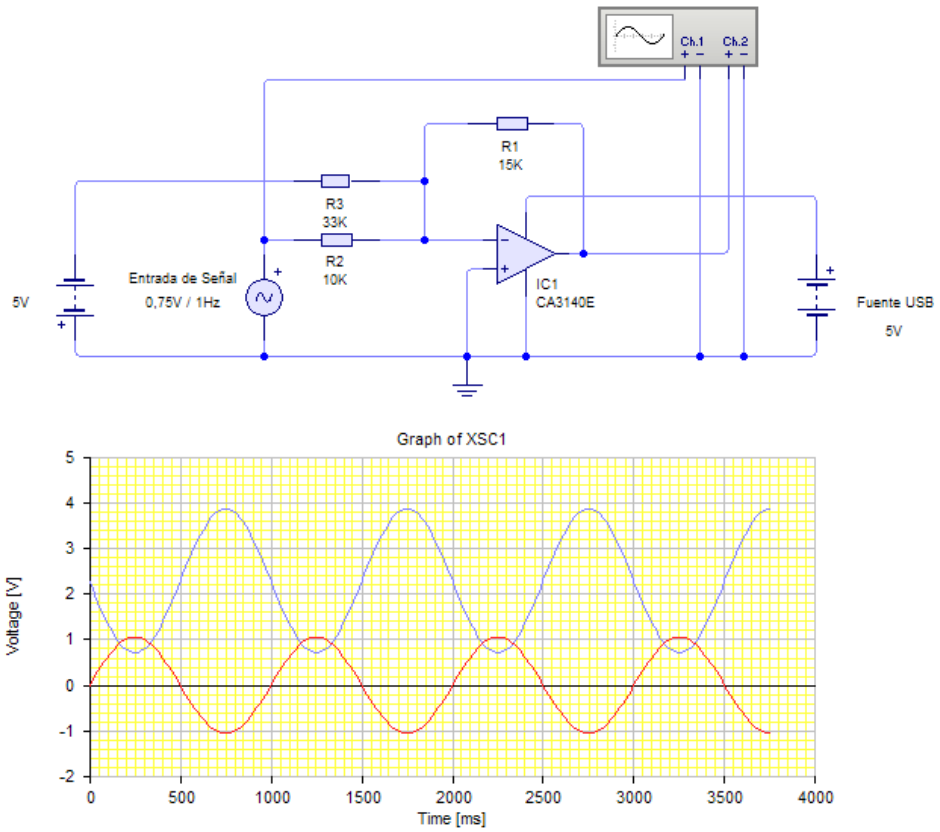

Fig10 – Diagrama de la placa de amplificación y corrimiento de offset

Este circuito se emplea por cada canal analógico que tenga el proyecto, en este caso son dos como se puede ver en la figura 11.

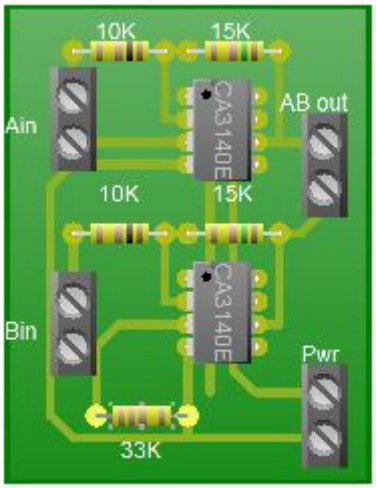

Fig11 – PCB de la placa de amplificación y corrimiento de offset

FUENTE DE ALIMENTACIÓN

Se diseño una fuente de alimentación del tipo lineal para brindar tensiones estables y fijas.
El circuito utiliza +5 V y -5 V, los cuales se obtienen por medio de dos reguladores 7805 y 7905.
Además se agregaron tensiones de +12 V y -12 V para futuras ampliaciones o modificaciones.
El diagrama de la fuente se puede ver en la figura 12.

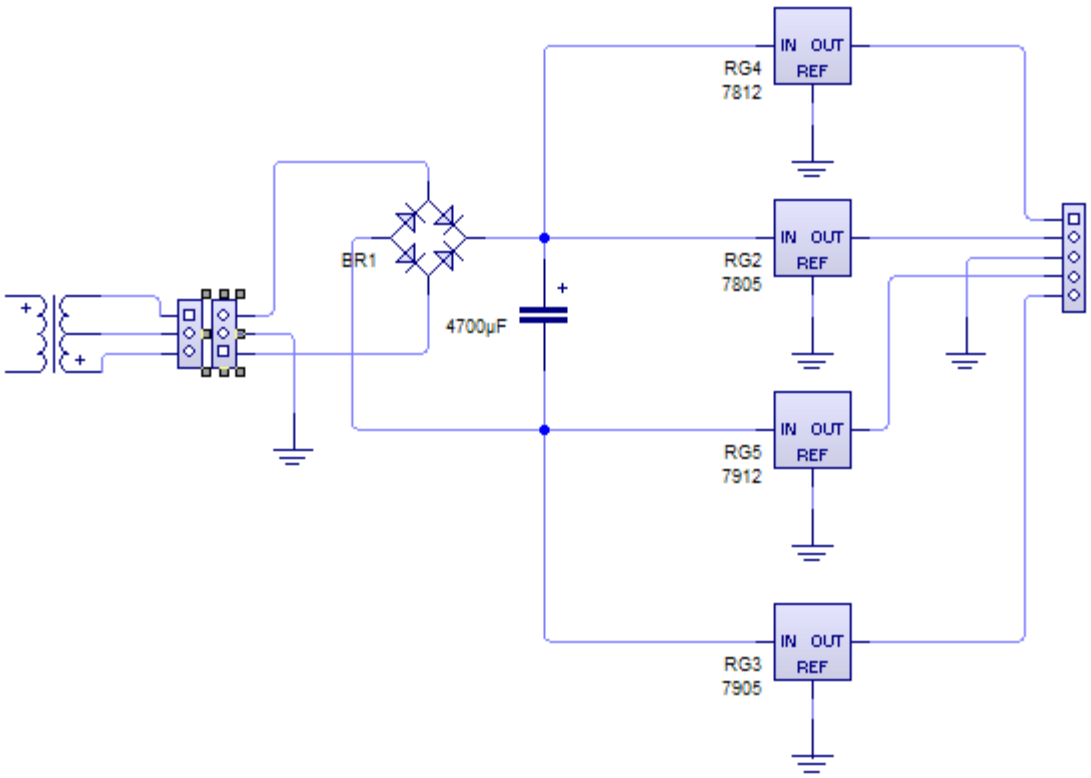

Fig12 – Diagrama de la placa de fuente de alimentación

RESULTADOS DE LAS PRUEBAS

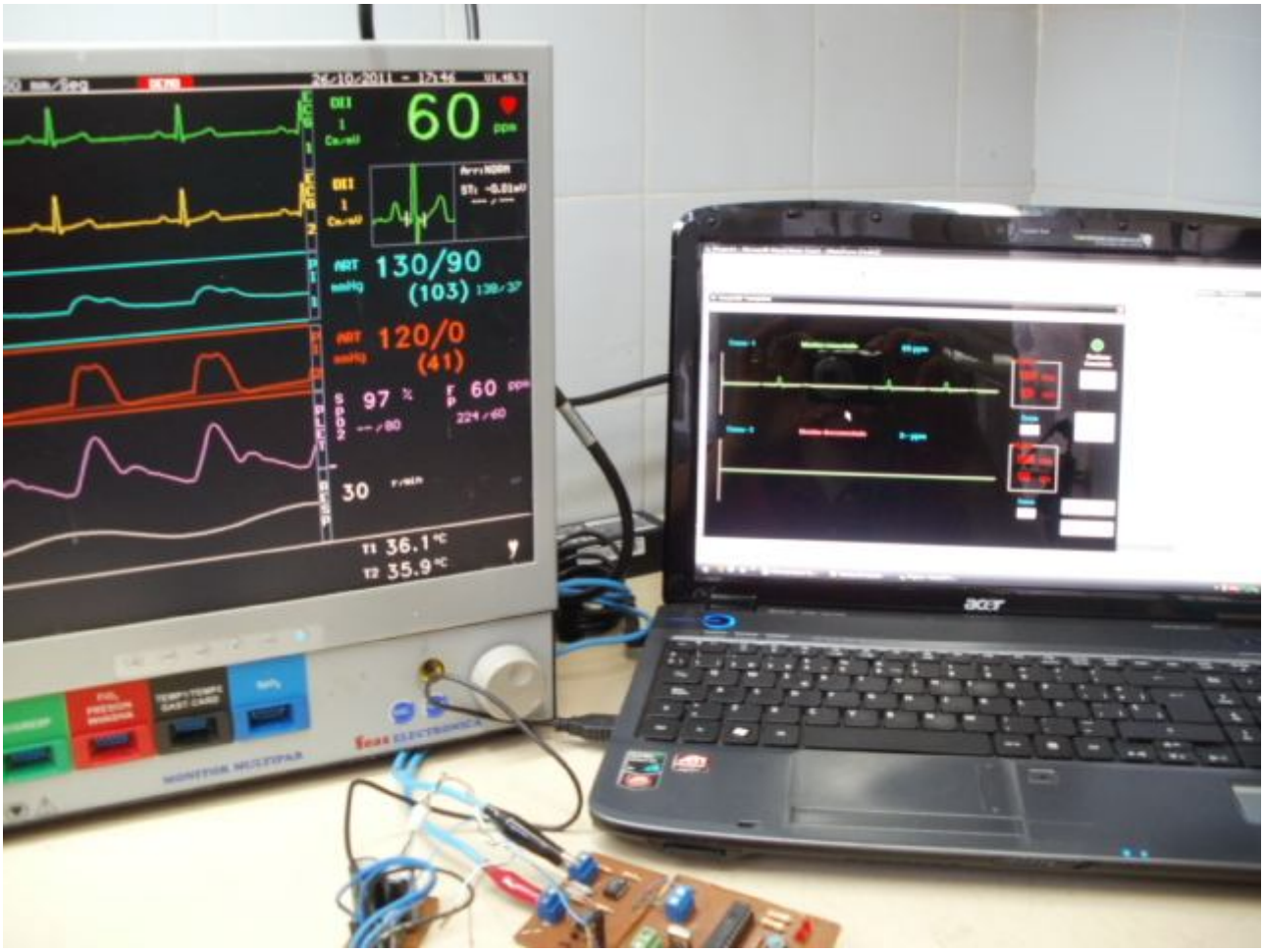

Fig13- imagen del proyecto conectado a la PC y al monitor Feas

Como puede verse en la imagen anterior, los resultados son los esperados en el diseño del proyecto.

CONCLUSIONES

En las pruebas de funcionamiento se tomaron lecturas electrocardiográficas con nivel de ruido en la señal, comparables con las de los monitores comerciales estándares.
A su vez con mínimos cambios se adapta a la medición de otros parámetros biomédicos de interés que este monitor soporte.
Es importante destacar también que el costo de producción del equipo es altamente inferior al de los equipos similares disponibles en plaza, y como su interfaz es compatible con cualquier PC hogareña, es posible inclusive su utilización para monitoreo en domicilio de pacientes o de bajos cuidados.
Se destaca además que todos los componentes utilizados son de bajo costo y amplia disponibilidad en el mercado local, asegurándose la factibilidad de mantenimiento del equipo.
Durante el desarrollo de este proyecto pudimos aplicar conocimientos que solo teníamos en teoría y sobretodo el utilizar una interfaz USB nos abrió las puertas hacia un nuevo horizonte en la comunicación con la PC (dejando de lado la comunicación por paralelo o por RS-232).
A lo largo del proyecto fuimos sorteando ciertos inconvenientes como ser el acondicionamiento de señal o la falta de disponibilidad de materiales e información técnica, ruido; sin embargo pudimos cumplir con nuestras expectativas al principio del trabajo dejando a futuro el suplir limitaciones con respecto a un equipo del mercado, enumeraremos algunas:

- Manejar un número mayor de camas.
- Una gama más amplia de parámetros que se encuentren disponibles en la ficha de salida a central (fig3).
- Ser compatible con un SO más estable y dedicado.
- Mantener registro de los parámetros tomados y de las alarmas.
- Datos personales del paciente.
- Impresión de señales.
- Mejoras en cuanto al diseño estético del equipo.
- Todo tipo de habilitaciones de acuerdo con las normalizaciones de la ANMAT.

ANEXOS:

LISTADOS DE PROGRAMAS

Programa cargado en el PIC18F2550

```
DEFINE OSC 48

' Define ADCIN parameters
Define      ADC_BITS       8  ' Set number of bits in result
Define      ADC_CLOCK      3  ' Set clock source (3=rc)
Define      ADC_SAMPLEUS  50  ' Set sampling time in uS
VariableADC0      var   byte    ' Create adval to store result
VariableADC1    var byte

USBBufferSizeMax   con 8  ' maximum buffer size
USBBufferSizeTX    con 8  ' input
USBBufferSizeRX    con 8  ' output

' the USB buffer...
USBBuffer        Var Byte[USBBufferSizeMax]
USBBufferCount   Var Byte

TRISA = %11111111 ' Set PORTA to all input
TRISB = 0
ADCON1 = %00001101      ' Set PORTA analog
Pause 500             ' Wait .5 second

' ************************************************************
' * main program loop - remember, you must keep the USB      *
' * connection alive with a call to USBService every couple  *
' * of milliseconds or so...                                 *
' ************************************************************
usbinit ' initialise USB...
ProgramStart:
   'gosub DoUSBIn
   ADCIN 0, variableadc0                    ' Read channel 0 to adval
   pause 5
   ADCIN 1, variableadc1
   pause 5
   usbbuffer[0]=variableadc0
   usbbuffer[1]=variableadc1
   if PORTA.2 = 1 then
   PORTB.7=1
   usbbuffer[2] = 255
   else
   PORTB.7=0
   usbbuffer[2]=0
   endif
   if PORTA.3 = 1 then
   PORTB.6=1
   usbbuffer[3] = 255
   else
   PORTB.6=0
   usbbuffer[3]=0
   endif

   gosub DoUSBOut

   goto ProgramStart
```

```
' ************************************************************
' * receive data from the USB bus                            *
' ************************************************************
DoUSBIn:
   USBBufferCount = USBBufferSizeRX              ' RX buffer size
   USBService                                    ' keep connection alive
   USBIn 1, USBBuffer, USBBufferCount, DoUSBIn   ' read data, if available
   if usbbuffer[0] = 10 then
   gosub dousbout
   endif
   return

' ************************************************************
' * wait for USB interface to attach                         *
' ************************************************************
DoUSBOut:
   USBBufferCount = USBBufferSizeTX              ' TX buffer siz
   USBService                                    ' keep connection alive
   USBOut 1, USBBuffer, USBBufferCount, DoUSBOut ' if bus available, transmit
data
   Return
```

Programa realizado en Visual Basic 6.0 cargado en la PC.

Si bien los valores del Product ID (PID) y Vendor ID (VID) puede ser modificados arbitrariamente por nosotros, dichos códigos son un tema muy delicado. La USB-IF(USB Implemmenters Forum, http://www.usb.org) tiene la autoridad (y responsabilidad) absoluta sobre la asignación de códigos VID únicos de 16 bits para cada fabricante que quiere comercializar sus productos USB. Esos códigos son obtenidos mediante una licencia y el pago de un impuesto (por una única vez) que oscila en los U$S 1500.Una vez que al fabricante se la ha asignado un código VID tiene a disposición 65.536 código PID codes (de 16-bits) para identificar únicamente distintos modelos de su creación. Como nuestro prototipo por el momento no va a ser comercializado, utilizamos los valores que venían en el programa por defecto.

El código es el siguiente:

```
'IDVendor y IdProduct para entablar comunicacion con el PIC
Private Const VendorID = 6017
Private Const ProductID = 2000
'variables para inicializar al inicio de windows
Private mReg As cQueryReg
Private Const cvRun As String =
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\"

'Variables para graficar en ambos canales, una recta entre dos puntos
'uno el punto anterior y el actual
Dim x1, y1, x2, y2, y12 As Integer
'Variable indica si silencian las alarmas
Dim silencio As Boolean
'Variable para contar, las pulsacioes por minutos
'Se cuentan las cantidad de datos que entran en un ciclo (delimitado por la
referencia)
'Sumador cuanta los datos, bandera marca inicio y el fin del ciclo y ppm es
'el calculo de las pulsaciones
Dim ppm1, ppm2, bandera, sumador, bandera2, sumador2 As Integer
'Variables que guardan el Zoom actual
Dim zoom1, zoom2 As Integer
'Variable que contiene la cantidad de segundos que lleva silencio = ON
```

```
Dim segundos As Integer
'Variables para saber si los canales estan conectados o no
Dim monitor1con, monitor2con As Boolean
'Variable para el ajuste automatico para la referencia, para contar las ppm
Dim pico1, pico2 As Integer

'Vectores que contienen la cantidad de bytes a enviar, y los datos
Private Const BufferInSize = 8
Private Const BufferOutSize = 8
Dim BufferIn(0 To BufferInSize) As Byte
Dim BufferOut(0 To BufferOutSize) As Byte


Private Sub al1max_KeyPress(KeyAscii As Integer)
'Filtro las pulsaciones de tecla (Ascii) para permitir solo numeros
    If (KeyAscii < 48 Or KeyAscii > 57) Then
        If (KeyAscii <> 127 And KeyAscii <> 8) Then KeyAscii = 0
    Else
        If Len(al1max) > 2 Then KeyAscii = 0
    End If
End Sub

Private Sub al1min_KeyPress(KeyAscii As Integer)
'Filtro las pulsaciones de tecla (Ascii) para permitir solo numeros
    If (KeyAscii < 48 Or KeyAscii > 57) Then
        If (KeyAscii <> 127 And KeyAscii <> 8) Then KeyAscii = 0
    Else
        If Len(al1min) > 2 Then KeyAscii = 0
    End If
End Sub

Private Sub al2max_KeyPress(KeyAscii As Integer)
'Filtro las pulsaciones de tecla (Ascii) para permitir solo numeros
    If (KeyAscii < 48 Or KeyAscii > 57) Then
        If (KeyAscii <> 127 And KeyAscii <> 8) Then KeyAscii = 0
    Else
        If Len(al2max) > 2 Then KeyAscii = 0
    End If
End Sub

Private Sub al2min_KeyPress(KeyAscii As Integer)
'Filtro las pulsaciones de tecla (Ascii) para permitir solo numeros
    If (KeyAscii < 48 Or KeyAscii > 57) Then
        If (KeyAscii <> 127 And KeyAscii <> 8) Then KeyAscii = 0
    Else
        If Len(al2min) > 2 Then KeyAscii = 0
    End If
End Sub

Private Sub cambiaroffset1_Change()
'Subrutina para redimensionar el offset, segun el scroll.
'Se multiplica por 10 para tener mayor variacion ante un cambio
offset1.y1 = cambiaroffset1.Value * 10
offset1.y2 = offset1.y1
End Sub

Private Sub cambiaroffset2_Change()
'Subrutina para redimensionar el offset, segun el scroll.
'Se multiplica por 10 para tener mayor variacion ante un cambio
offset2.y1 = cambiaroffset2.Value * 10
offset2.y2 = offset2.y1
```

```
End Sub

Private Sub cambiarref1_Change()
'Subrutina para redimensionar la referencia, segun el scroll.
'Se multiplica por 10 para tener mayor variacion ante un cambio
ref1.y1 = cambiarref1.Value * 10
ref1.y2 = ref1.y1
End Sub

Private Sub cambiarref2_Change()
'Subrutina para redimensionar la referencia, segun el scroll.
'Se multiplica por 10 para tener mayor variacion ante un cambio
ref2.y1 = cambiarref2.Value * 10
ref2.y2 = ref2.y1
End Sub

Private Sub Command1_Click()
'Cuando se silencia, se activa el silencio y se resetea el contador de segundos
'que comanda el timer2, a los 180 segundos debe ponerse silencio = OFF
silencio = True
segundos = 0
End Sub

Private Sub Command2_Click()
'Si la instruccion es desconectar, desconecta la comunciacion USB, sino la
conecta
If Command2.Caption = "Desconectar" Then
DisconnectFromHID
      'Pone todos los carteles indicadores y variables de control respecto de la
conexion
      Shape1.FillColor = &HFF&
      Command2.Caption = "Conectar"
      Label1 = "No Conectado"
      Label4 = "Monitor No Conectado"
      Label4.ForeColor = &H8080FF
      monitor2con = False
      Label2 = "Monitor No Conectado"
      Label2.ForeColor = &H8080FF
      monitor1con = False
Else
ConnectToHID (Me.hwnd)
      Shape1.FillColor = &HC000&
      Command2.Caption = "Desconectar"
      Label1 = "Conectado"
End If
End Sub

Private Sub Command3_Click()
If Command3.Caption = "Habilitar" Then
Command3.Caption = "Deshabilitar"
Command3.BackColor = 12648384
ppml1.Visible = True
Else
Command3.Caption = "Habilitar"
Command3.BackColor = 8421631
ppml1.Visible = False
End If
End Sub

Private Sub Command4_Click()
If Command4.Caption = "Habilitar" Then
```

```
Command4.Caption = "Deshabilitar"
Command4.BackColor = 12648384
ppml2.Visible = True
Else
Command4.Caption = "Habilitar"
Command4.BackColor = 8421631
ppml2.Visible = False
End If
End Sub

' ****************************************************************
' when the form loads, connect to the HID controller - pass
' the form window handle so that you can receive notification
' events...
'****************************************************************
Private Sub Form_Load()
   ' Algoritmo para setear el registro, para inicializar con window
automaticamente
   Set mReg = New cQueryReg
   Dim s As String
   ' Poner la clave
   s = mReg.GetRegString(cvRun, App.EXEName)
   If s = "" Then
    If mReg.SetReg(cvRun, App.EXEName, App.Path + "\" + App.EXEName + ".exe") <>
ERROR_NONE Then
        MsgBox "ERROR al crear la clave de Inicio"
    End If
   End If

   'Pasa la informacion del programa para poder entablar una comunicacion USB
   ConnectToHID (Me.hwnd)
   'Pone los indicadores de que las camas no estan conectadas
   Label4 = "Monitor No Conectado"
   Label2 = Label4
   Label4.ForeColor = &H8080FF
   Label2.ForeColor = &H8080FF
   monitor1con = False
   monitor2con = False

   'Se redefine el zoom de cada cama, segun el valor de su respectivo control
ComboBox
   zoom1 = Int(Mid(z1, 2)) * 3
   zoom2 = Int(Mid(z2, 2)) * 3

   'Reseteo de la variable para controlar el tiempo de silencio de alarmas
   segundos = 0

   'Se resetean todas las variables para el calculo de las pulsaciones por
minuto
   bandera = 0
   sumador = 0
   bandera2 = 0
   sumador2 = 0
   ppm1 = 0
   ppm2 = 0

   'Silencio desactivado
   silencio = False

   'Reseteo de las variables para la grafica de la señal
   x1 = 0
```

```
   x2 = 1
   y1 = 0
   y2 = 0
   y22 = 0

   'Se definen la linea de referencia y de offset, como valor inicial
   'El offset es el eje y=0 y la referencia es la mitad del eje positivo
   'Se hacen invisibles todas las lineas y Scrolls para solo acceder desde el
menu
   pico1 = 0
   offset1.Visible = False
   pico2 = 0
   offset2.Visible = False
End Sub

Private Sub Form_Unload(Cancel As Integer)
   'Cuando se cierra el programa, se desconecta la comunicacion USB
   DisconnectFromHID
      'Pone los carteles y variables, a la posicion de no conectado
      Label4 = "Monitor No Conectado"
      Label4.ForeColor = &H8080FF
      monitor2con = False
      Label2 = "Monitor No Conectado"
      Label2.ForeColor = &H8080FF
      monitor1con = False
End Sub

Public Sub OnPlugged(ByVal pHandle As Long)
'Funcion a la que se accede cuando un dispositivo USB es conectado
'Corroboramos que conincida nuestro VendorID y ProductID con el del hardware
   If hidGetVendorID(pHandle) = VendorID And hidGetProductID(pHandle) =
ProductID Then
      'Se crean indicaciones visiules de que esta conectado
      Label1 = "Conectado"
      Shape1.FillColor = &HC000&
      Command2.Caption = "Desconectar"
   End If
End Sub

Public Sub OnUnplugged(ByVal pHandle As Long)
'se verifica que el dispositivo que se desconecto conincida con nuestro hardware
   If hidGetVendorID(pHandle) = VendorID And hidGetProductID(pHandle) =
ProductID Then
      'Indicaciones visuales de que esta desconectado
      Label1 = "No Conectado"
      Shape1.FillColor = &HFF&
      Command2.Caption = "Conectar"
   End If
End Sub

Public Sub OnChanged()
   'Rutina a la que accede si hay algun datos para leer
   Dim DeviceHandle As Long

   DeviceHandle = hidGetHandle(VendorID, ProductID)
   hidSetReadNotify DeviceHandle, True
End Sub

Public Sub OnRead(ByVal pHandle As Long)
'Se lee el vector de datos, indicando cual es el primer byte a leer
'hidRead devuelve 1 si existian datos para leer
```

```
   If hidRead(pHandle, BufferIn(0)) Then
      'En buffer(3) y buffer(4) se obtiene un 255 si hay conectado el canal
      'y 0 si no hay nada conectado
      'segun esos valores se determinan los carteles y variables de conectado
      'de cada canal (cama)
      If BufferIn(3) > 1 Then
      Label4 = "Monitor Conectado"
      Label4.ForeColor = &H80FF80
      monitor2con = True
      Else
      Label4 = "Monitor No Conectado"
      Label4.ForeColor = &H8080FF
      monitor2con = False
      End If

      If BufferIn(4) > 1 Then
      Label2 = "Monitor Conectado"
      Label2.ForeColor = &H80FF80
      monitor1con = True
      Else
      Label2 = "Monitor No Conectado"
      Label2.ForeColor = &H8080FF
      monitor1con = False
      End If

      'Variable auxiliar para facilitar la grafica de datos
      Dim i As Integer

      'Con esta subrutina, se realiza un borrado haca adelante, para ir
actualizando
      'la grafica, para graficar en tiempo real

            'Si el valor a escribir es el primero, se redibujan los ejes
verticales, para
      'impedir un posible borrado
      If x1 = 0 Then
          vert1.Refresh
          vert2.Refresh
      End If

      'Si el valor a escribir es el ultimo, se resetean el contador x1 y x2
      'que indican la posicion horizontal, del dato actual.
      If x2 = 250 Then
       x1 = 0
       x2 = 0
      End If

      x1 = x2
      x2 = x2 + 1

      For i = 0 To 20
         Picture1.Line (x2 * 20 + i, vert1.y1)-(x2 * 20 + i, vert1.y2)
         Picture1.Line (x2 * 20 + i, vert2.y1)-(x2 * 20 + i, vert2.y2)
         'Picture1.PSet (x2 * 20 + i, hor1.y1), vbWhite
         'Picture1.PSet (x2 * 20 + i, hor2.y2), vbWhite
      Next



      'Se grafica una recta, entre el punto anterior y el punto actual, para dar
sensación
```

```
      'de continuidad, si la cama esta habilitada
      'para ello se monta sobre un offset y se multiplica por el zoom
      'al control picture.line hay que pasarle como parámetros (x1,y1) y (x2,y2)
      'de la recta



      If Command3.Caption = "Deshabilitar" Then

      'Se declara la variable graficay1 para limitar el valor máximo o mínimo
que puede tomar
      Dim graficay1 As Integer

      graficay1 = (BufferIn(2) - 115) * zoom1 + offset1.y1
      If graficay1 < vert1.y1 Then graficay1 = vert1.y1
      If graficay1 > vert1.y2 Then graficay1 = vert1.y2 - 5

      Picture1.Line (x1 * 20, y1 * zoom1 + offset1.y1)-(x2 * 20, graficay1),
vbGreen '115 nivel de cont

      y1 = (graficay1 - offset1.y1) / zoom1

      'se actualiza que ingresaron datos
      'la cantidad de datos por ciclo nos da las pulsaciones por minuto

      sumador = sumador + 1

      If pico1 < (BufferIn(2) - 115) * zoom1 + offset1.y1 Then
       pico1 = (BufferIn(2) - 115) * zoom1 + offset1.y1
       ref1.y1 = pico1 - 20
       ref1.y2 = ref1.y1
       ref1.Refresh
      End If

      'Se realiza un algoritmo para contar cuantos datos se han leído en el
ciclo
      'delimitado por la línea de referencia
      'Para ello la variable bandera incica el inicio y el fin del ciclo
      'y luego se utiliza un contador (sumador)
      'Una vez finalizado el conteo, sabiendo que ingresan datos cada 10ms
(tiempo del PIC)
      'se realiza el calculo de las ppm= 20/sumador *60
      'luego se escribe en el control corespondiente para su visualizacion

      If ((offset1.y1 + y1 * zoom1) < ref1.y1) And (bandera = 0) Then
        bandera = 1
        If sumador <> 0 Then
            ppm1 = Int(((100 / (sumador)) * 60))
            ppml1 = ppm1 & " ppm"
        End If
        sumador = 0
       End If
     If ((offset1.y1 + y1 * zoom1) > ref1.y1) And (bandera = 1) Then
            bandera = 0
     End If

     End If

     ' se realiza el mismo prodecimiento para el canal 2
      If Command4.Caption = "Deshabilitar" Then
```

```
      Dim graficay2 As Integer
      'Se limita el valor maximo y minimo que peude graficar
      graficay2 = (BufferIn(1) - 125) * zoom2 + offset2.y1
      If graficay2 < vert2.y1 Then graficay2 = vert2.y1
      If graficay2 > vert2.y2 Then graficay2 = vert2.y2 - 5

      Picture1.Line (x1 * 20, y12 * zoom2 + offset2.y1)-(x2 * 20, graficay2),
vbGreen '113 nivel de cont

      sumador2 = sumador2 + 1

      'Se actualizan la posicion horizontal anterior y la actual

      y12 = (graficay2 - offset2.y1) / zoom2

      If pico2 > (BufferIn(1) - 125) * zoom2 + offset2.y1 Then
       pico2 = (BufferIn(1) - 125) * zoom2 + offset1.y1
       ref2.y1 = pico2 - 20
       ref2.y2 = ref2.y1
      End If

        'el mismo metodo para el canal1
      If ((offset2.y1 + y12 * zoom2) < ref2.y1) And (bandera2 = 0) Then
        bandera2 = 1
        If sumador2 <> 0 Then
            ppm2 = Int(((100 / (sumador2)) * 60))
            ppml2 = ppm2 & " ppm"
        End If
        sumador2 = 0
      End If
      If ((offset2.y1 + y12 * zoom2) > ref2.y1) And (bandera2 = 1) Then
        bandera2 = 0
      End If

      End If

   End If
End Sub


Public Sub WriteSomeData()
'Instrucciones para enviar datos al PIC, no lo utilizamos
'   BufferOut(0) = 0
'   BufferOut(1) = 10
'   hidWriteEx VendorID, ProductID, BufferOut(0)
End Sub

Private Sub sr2_Click()
'Algoritmo para seteo de los niveles de referencia, se bloquea en estado de
funcionamiento
'para prevenir el ingreso de datos, y se desbloquea al entrar desde el menu para
configurar
If sr2.Caption = "Finalizar" Then
cambiarref2.Visible = False
sr2.Caption = "Seleccionar Referencia"
ref2.Visible = False
Else
cambiarref2.Visible = True
ref2.Visible = True
sr2.Caption = "Finalizar"
End If
```

```
End Sub

Private Sub Timer1_Timer()
Dim alarma As Boolean
'Si esta conectado,El timer maneja una variable, alarma, para setearla si las
ppm
'estan dentro de los niveles de alarma maxima o minima

If monitor1con And Command3.Caption = "Deshabilitar" Then
    If ppm1 < Int(al1min) Then
        al1min.BorderStyle = Int(al1min.BorderStyle + 1) Mod 2
        alarma = True
    Else
        al1min.BorderStyle = 0
    End If

    If ppm1 > Int(al1max) Then
        al1max.BorderStyle = Int(al1max.BorderStyle + 1) Mod 2
        alarma = True
    Else
        al1max.BorderStyle = 0
    End If
End If

'lo mismo para el canal 2
If monitor2con And Command4.Caption = "Deshabilitar" Then
    If ppm2 < Int(al2min) Then
        al2min.BorderStyle = Int(al2min.BorderStyle + 1) Mod 2
        alarma = True
    Else
        al2min.BorderStyle = 0
    End If

    If ppm2 > Int(al2max) Then
        al2max.BorderStyle = Int(al2max.BorderStyle + 1) Mod 2
        alarma = True
    Else
        al2max.BorderStyle = 0
    End If
End If

'Si la variable fue seteada y no esta activado el silencio, se emite un sonido
    If alarma And Not (silencio) Then
        Beep 1000, 100
        alarma = False
    End If

End Sub


Private Sub Timer2_Timer()
'Controla que cada 100 segundos, se reponga el sonido de la alarma
segundos = segundos + 1
If segundos = 100 Then
silencio = False
segundos = 0
End If
End Sub

Private Sub z1_Click()
'Al cambiar el control correspondiente al zoom, se actualiza a su valor elegido
```

```
zoom1 = Int(Mid(z1, 2)) * 3
pico1 = 0
End Sub

Private Sub z2_Click()
'Al cambiar el control correspondiente al zoom, se actualiza a su valor elegido
zoom2 = Int(Mid(z2, 2)) * 3
pico2 = 0
End Sub
```

FOTOS DEL PROTOTIPO

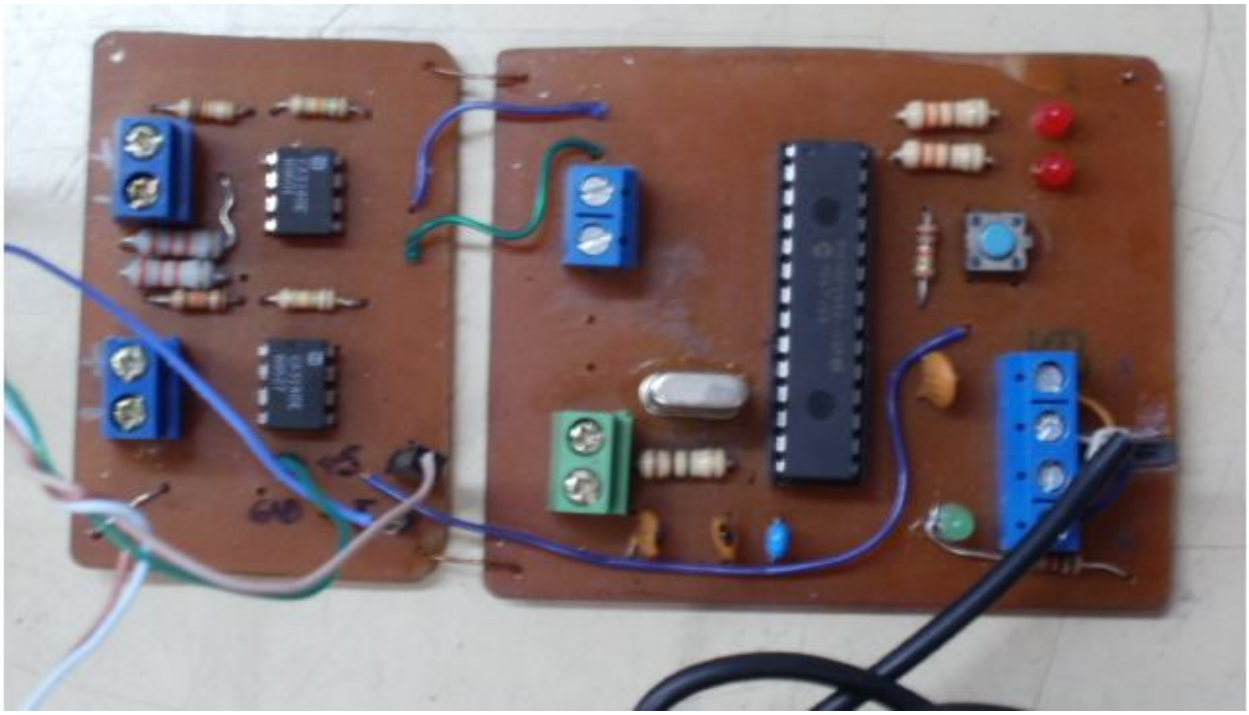

Fig14- Imagen del proyecto

En esta imagen puede verse plasmado el diagrama anteriormente propuesto

FOTOS DE PANTALLA

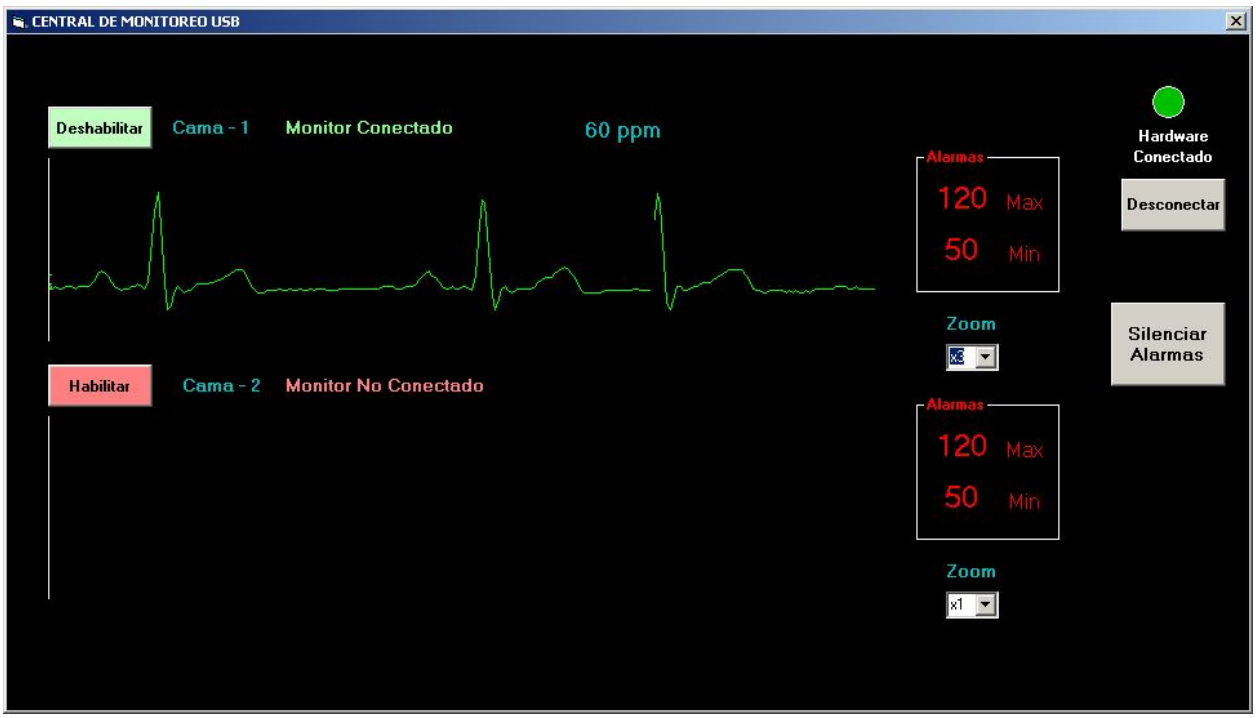

Fig15- Captura de pantalla del programa en VB corriendo en la PC

COMPATIBILIDAD ELECTROMAGNÉTICA

Para realizar el interconexionado entre el monitor y el hardware desarrollado, se realizo una investigación de la normas de EMC (compatibilidad electromagnética).
Como definición, un equipo con compatibilidad electromagnética tiene "la capacidad para funcionar de forma satisfactoria en su entorno electromagnético sin provocar perturbaciones electromagnéticas sobre cualquier cosa de ese entorno".
La compatibilidad electromagnética se ocupa de dos problemas diferentes, que dan lugar a dos ramas de la misma:
• Inmunidad o Susceptibilidad Electromagnética (EMS): Ese aparato, equipo o sistema debe ser capaz de operar adecuadamente en ese entorno sin ser interferido por otros.
• Interferencias Electromagnéticas (EMI): No debe ser fuente de interferencias que afecten a otros equipos de ese entorno (emisiones electromagnéticas).
Por lo tanto, para que satisfaga algunas cuestiones de la EMC, debe cumplir aspectos del EMS y EMI. Como el hardware realizado prácticamente no es fuente de interferencias electromagnéticas (EMI), nos centramos en estudiar aspectos de su susceptibilidad (EMS)

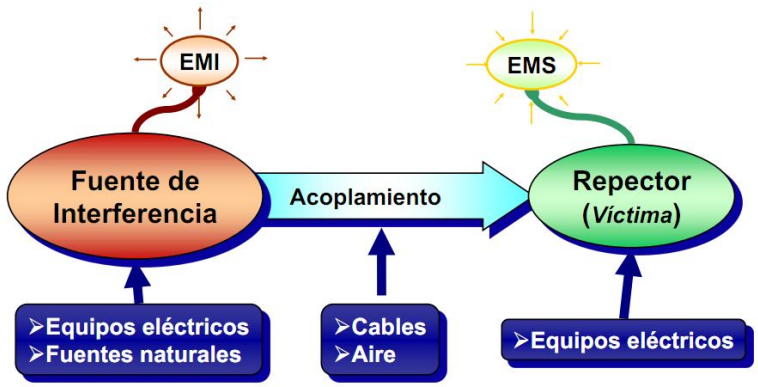

Fig16- EMC

Como concepto principal, el ruido genera interferencia, que es un efecto no deseado Por interferencia se entiende cuando el ruido provoca un mal funcionamiento de un equipo.
El ruido NO PUEDE ELIMINARSE, sólo puede REDUCIRSE.
Nuestro hardware debe tomar la señal del monitor, sin interferencias, por lo que centraremos el estudio en reducir los efectos en la señal.
Debido a que se instalara en un cuarto de terapia intensiva, se descarta el uso de sistemas de comunicación inalámbricos, ya que posee cierta emisión de interferencias, y es un modo de comunicación muy susceptible, respecto a equipamientos de la terapia (rayos x, equipamiento electrónico con fuentes conmutadas), etc. Además, es un lugar donde las paredes están plomadas lo que dificulta más aun la comunicación inalámbrica.
Por esta razón seleccionamos como medio de comunicación cables (de ahora en adelante, medio de acoplamiento ya que lo vemos desde un punto de vista del ruido).
Para detectar que aspectos debemos tener en cuenta a la hora de interconectar el monitor a nuestro equipo, debemos identificar las distintas fuentes de ruido.

FUENTES DE RUIDO
Ruido intrínseco (propio del hardware realizado): En etapas de BAJA SEÑAL realizamos una estructura con AO NO INVERSORA ya que proporciona mejor relación Señal/Ruido que la

estructura INVERSORA, por lo que consideramos que el hardware se insensibilizo respecto a este tipo de ruidos.

Ruido extrínseco: Algunas posibles fuentes de ruido extrínseco presentes en la sala a trabajar

Transitorios de conmutación en circuitos RLC
Arcos en contactos de relés y contactores
Bucles de masa y cableados deficientes
Perturbaciones en la alimentación de red
Máquinas eléctricas: transformadores, motores
Fluorescentes, circuitos eléctricos trabajando en conmutación
Equipos de Electrónica de Potencia trabajando en conmutación
Emisoras de RADIO, TV, RADAR
Perturbaciones naturales (descargas eléctricas, etc) - No tenido en cuenta

El panorama general del estudio que debemos realizar.

Fig17-Fuentes de ruido

En el medio de acoplamiento, se pueden dar dos tipos de interferencias:

Interferencias conducidas, que son perturbaciones transmitidas a través de los conductores

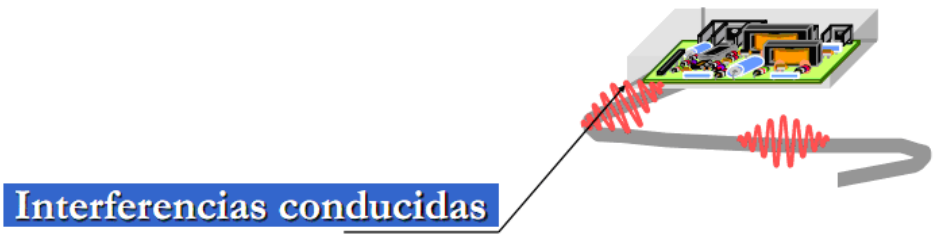

Fig18-Interferencias conducidas

Interferencias Radiadas: que provienen de campos electromagnéticos, y se propagan a través de acoplamientos capacitivos e inductivos

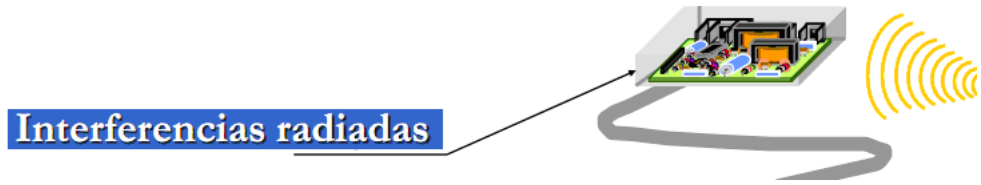

Fig19-Interferencias radiadas

Interferencias radiadas
La señal que debemos medir, es de baja potencia y baja frecuencia, por lo que tenemos que tener en cuenta el efecto de la radiación electromagnética presente en el ambiente (por distintas causas), que pueden llegar a afectar a la señal original. El campo electromagnético se acopla a la señal por efectos capacitivos e inductivos, por lo que se estudiara el efecto que tiene el efecto capacitivo, en baja señal.

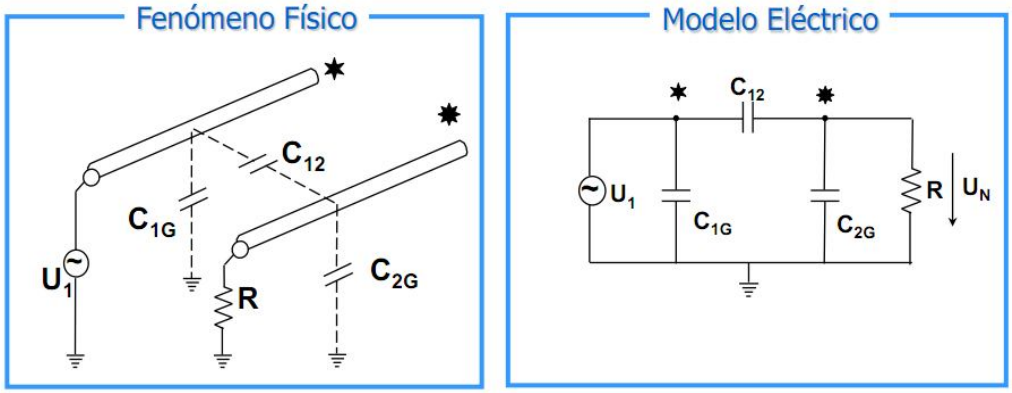

Fig20-distribución física del conductor

La distribución física del conductor, genera naturalmente efectos capacitivos, que facilitan que se acople radiación electromagnética, como solución a este efecto, se puede realizar un apantallamiento del conductor, cuyo efecto es el siguiente

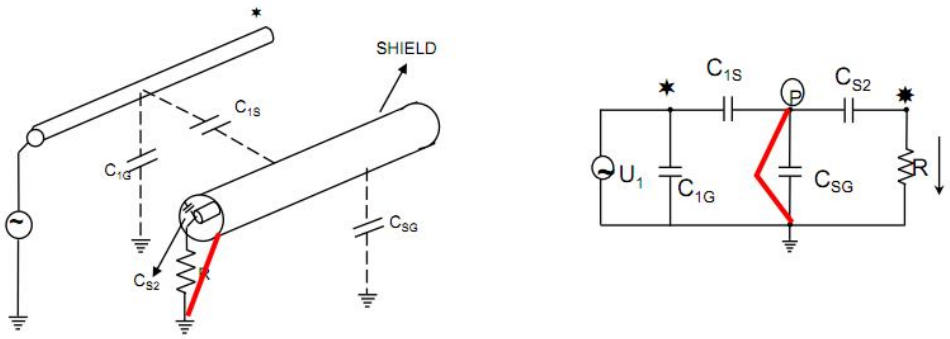

Fig21-Apantallamiento

El apantallamiento (por lo general de aluminio o un mallado) reduce la interferencia electroestática, si se pone a tierra un solo extremo de la pantalla. También para reducir más la interferencia, se debe minimizar la longitud del extremo saliente del conductor sobre la pantalla (de lo contrario, produce un efecto capacitivo y genera una tensión inducida)

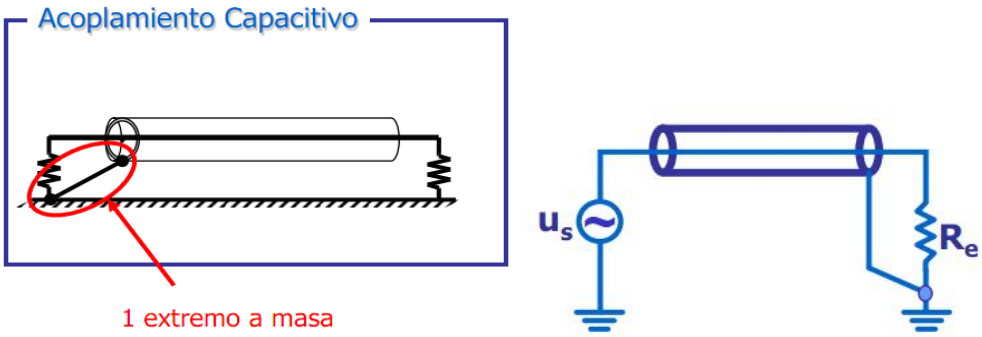

Fig22-Acoplamiento capacitivo

También, para obtener menor interferencia a altas frecuencias, se puede disponer de un esquema como el que sigue.

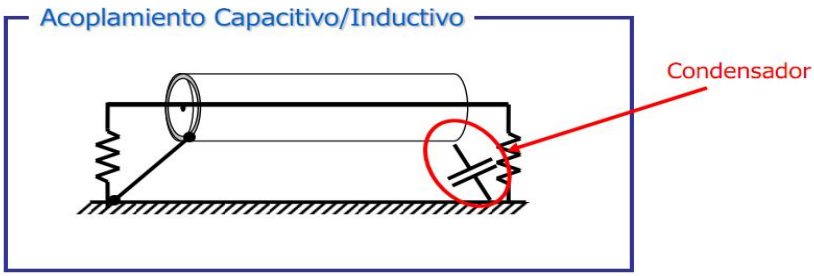

Fig23-Acoplamiento capacitivo/inductivo

Ya que la alta frecuencia genera interferencia en acoplamientos inductivos, se puede poner un capacitor al otro extremo. Por lo general la fuente más importante de ruido es la de 50Hz, proveniente de tubos, etc, por lo que realizaremos un esquema de conexionado correspondiente a reducir interferencias a baja frecuencia.

La pantalla con los dos lados a masa no tiene ningún efecto en bajas fecuencias sobre la interferencia y puede provocar ruido por lazo de masa.

Fig24-Lazo de masa

Como recomendación, las normas se detalla para que un equipo sea electromagnéticamente compatible (EMC), el apantallado debe conectarse a masa del lado de la señal y no del lado del amplificador, siendo este el mejor esquema de acuerdo a nuestras condiciones que se plantearon inicialmente.

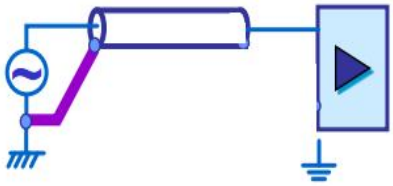

Fig25-Apantallamiento

Interferencias conducidas.
Estas interferencias, se dan por un mal conexionado en la tierra de los equipos, provocando ruidos de modo común. Para la conexión de la tierra, la norma establece que si existen equipos electrónicos digitales (con fuente conmutada), debe tomarse como si fueran equipos de alta frecuencia, por lo que el conexionado se realiza según el siguiente esquema:

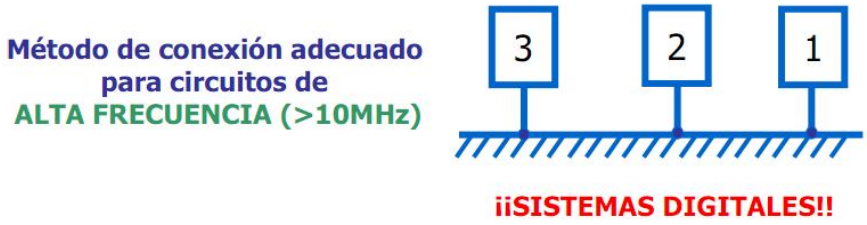

Fig26-Interferencias conducidas

Donde la tierra es un sistema de instalación mallado de varios puntos sobre el piso de la sala.
Esto podría resultar en algún inconveniente, ya que la actual disposición de la tierra de la sala corresponde al siguiente esquema, que es el recomendado para bajas frecuencias, pero que no sean equipos digitales.

Fig27-Conexión de baja frecuencia

El monitor tiene la salida de señal referida a tierra, y nuestro hardware también está conectado a tierra, ya que el GND de la PC, está referida a tierra, siendo los cables de tierra bastante largo hasta llegar a la jabalina, por lo que tenemos una posible gran señal en modo común, en el interconexionado de los equipos. Realizando una simplificación, tenemos lo siguiente:

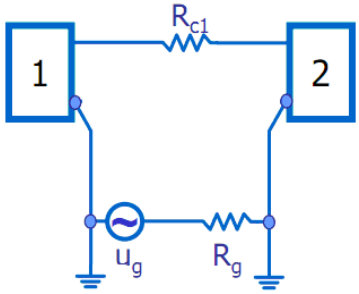

Fig28-Referencias de Tierra

Donde Rc1 es la resistencia del cable, y Ug es la diferencia de tensión que aparece entre las tierras de los equipos y Rg la resistencia entre masas. En este circuito se ve claramente que aparece una tensión en modo común, que podría ser hasta un volt, y perderíamos la señal original.
Una solución, podría haber sido, aislar el operacional o la señal, para que la señal sea flotante, y así, el ruido aparecería en modo diferencial, pero deberíamos haber puesto transformadores de aislamiento, o en su defecto enviar la señal en forma digital, etc.

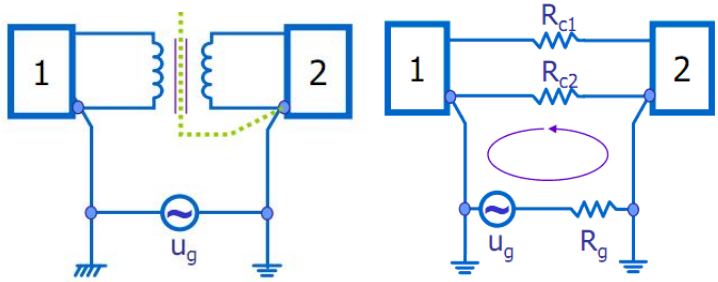

Fig29-Referencias de Tierra

Pero para reducir este efecto, usamos un segundo cable, para unir las tierras, disponiendo del siguiente esquema:

El segundo cable, funcionaria como cable de guardia, reduciendo en un gran porcentaje el tensión en modo común, ya que ahora la tensión en modo común que aparece es la que circula por Rc2, cuya resistencia es muy pequeña.
Esta solución, es la más sencilla y barata de implementar, ya que se ve reducida la tensión en modo común, y en relación a la amplitud de la señal, no genera interferencia apreciable.
De acuerdo a lo estudiado, se concluye que utilizaremos, cable con dos hilos conductores, mallado, para la toma de la señal. De un hilo se tomara la señal, el restante se utiliza como cable de guardia y la malla para realizar el apantallamiento.

PERTUBACIONES DE RED
Las perturbaciones en la red, provocan perturbaciones en la alimentación de los equipos. En la siguiente tabla se dan ejemplos de posibles perturbaciones:

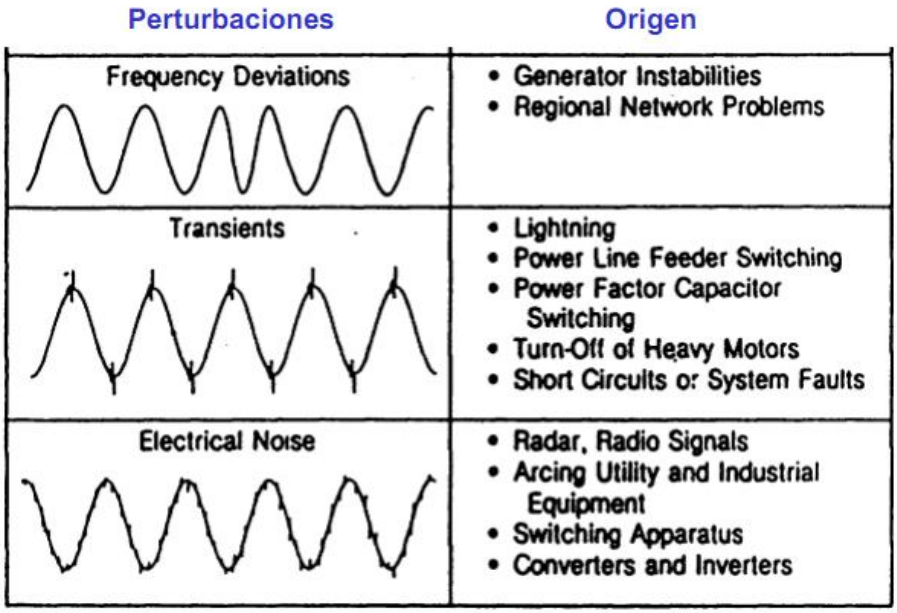

Fig30-Perturbaciones de Red

PRUEBAS REALIZADAS AL EQUIPO

En nuestro taller, una vez finalizado el prototipo se procede a verificar su la inmunidad al ruido. Como prueba utilizamos una buena fuente de ruido: intercalamos en la línea de la red de alimentación de 220Vac, un transformador con un gran consumo, en este caso utilizamos un gran trafo alimentando una lámpara dicroica, y una lima de metal. Con un buen aislamiento realizamos una pasada del electrodo por la lima, generando y sometiendo al equipo a variaciones en la red eléctrica, y ruido desde las bobinas del transformador. El equipo debe estar cerca del transformador y además conectado a la misma línea,

El esquema es el siguiente:

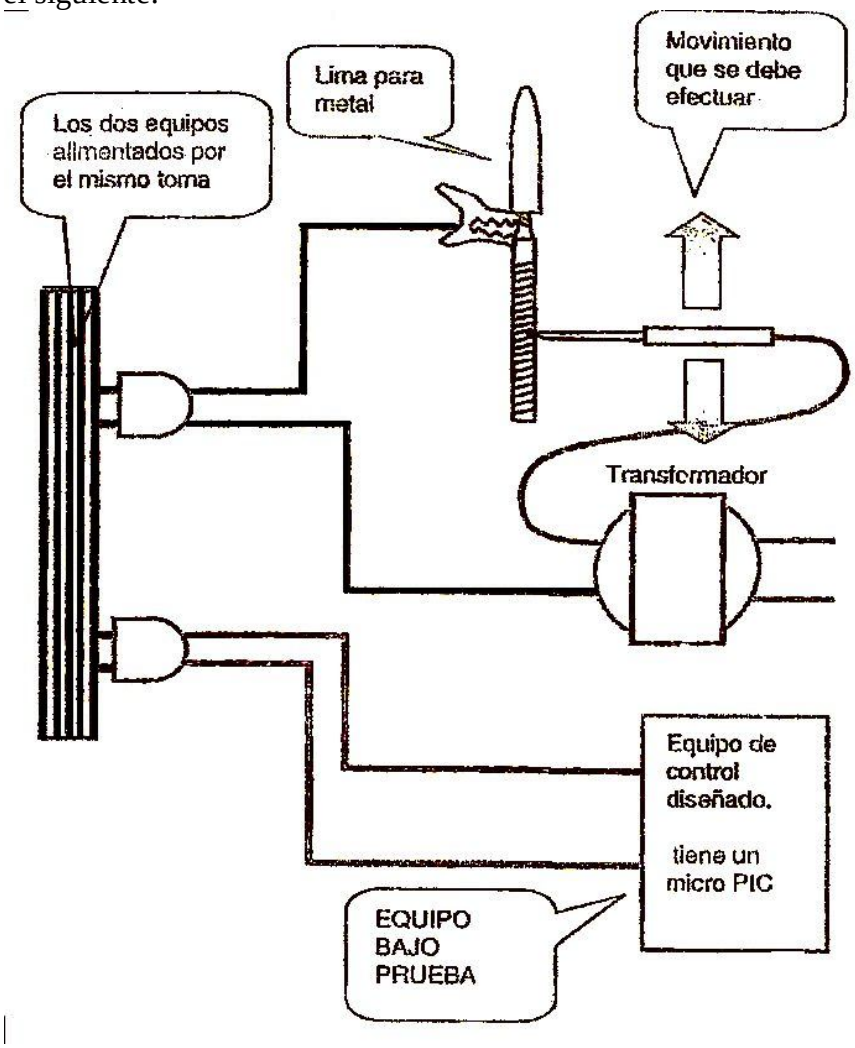

Fig31- circuito para probar la incidencia del ruido en el equipo

Una vez ensayado, se observo que el microcontrolador se reiniciaba cuando el ruido se hacia presente es por eso, y teniendo en cuenta los resultados de opto por anexar a la fuente del equipo un filtro FI dando una performance excelente.

El diagrama circuital del mismo es el siguiente:

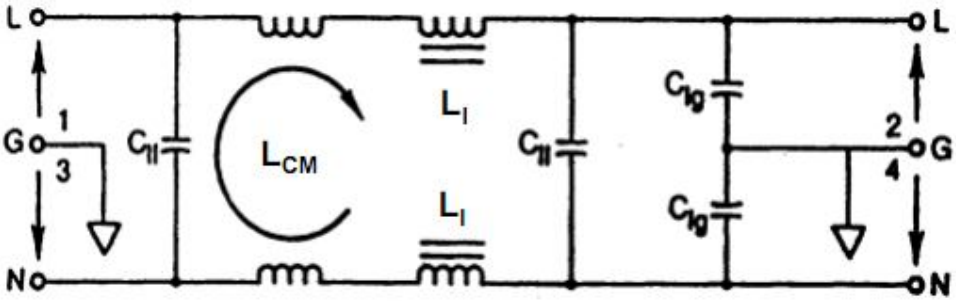

Fig32- Filtro FI

Fig33- Fotografía del filtro

De esta manera, el equipo diseñado es inmune a las oscilaciones de frecuencias bajas e intermedias de la red.