UNIVERSIDAD MAYOR DE SAN ANDRES
FACULTAD DE INGENIERIA
INGENIERIA ELECTRICA

PROGRAMACION PARA

INGENIERIA II

PROYECTO DE FINAL DE SEMESTRE

UNIVERSITARIOS	:	Lima Poma Ramiro Luis Choque Villca Henry J. Montaño Tinta Pedro Alex. Mollo Pari Claudia
DOCENTE	:	Lic. Irma Prado
CARRERA	:	Ingeniería Eléctrica
FECHA DE ENTREGA	:	Diciembre 9 del 2011

UNIVERSIDAD MAYOR DE SAN ANDRES

FACULTAD DE INGENIERIA

PROYECTO FINAL DE COMPUTACION PARA INGENIERIA II

1. OBJETIVOS

- El objetivo del presente proyecto es llevar los conocimientos en programación JAVA al control de puertos de una computadora, en sistema operativo Windows.

- Investigar formas posibles que se tiene para abrir los puertos de una computadora, evaluar cada una de ellas y encontrar la mejor solución para nuestro propósito.

- Determinar los materiales y programas que se deben necesitar, para el control del puerto.

- Investigar cuales son los procedimientos para usar el puerto USB de una computadora personal, ya que los nuevos modelos no incluyen puertos serie ni paralelo.

2. PROCEDIMIENTO

El fin del proyecto de fin de curso de la materia de COMPUTACION PARA INGENIERIA, en nuestro caso es realizar la apertura del puerto serie de una computadora personal, para controlar y realizar la automatización de una vivienda, donde se controlara el encendido y apagado del circuito de iluminación de la vivienda.

Para este propósito pensamos que la solución más actual que podemos proponer es una el puerto USB, el primer problema fue como realizar la gestión de este puerto y unir este con el lenguaje JAVA, los siguientes párrafos hacen referencia a los puntos investigados y usados en el proyecto.

2.1 PROTOCOLO USB

Sin realizar mucha teoría que se encuentra en todos lados de red, nos vamos directo al grano, lo que se realizara es una comunicación bidireccional usando el puerto USB, se realizara la gestión del mismo usando un microcontrolador de la familia MICROCHIP de código PIC18F4550.

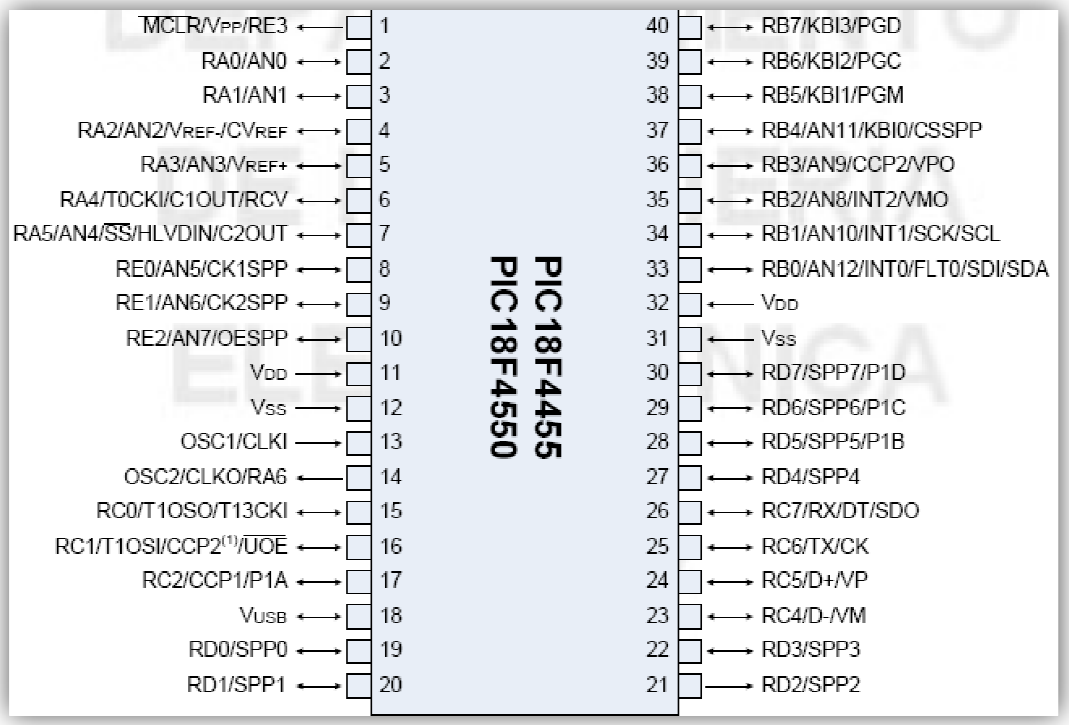

FIGURA Nº 1
ENCAPSULADO DEL MICROCONTROLADOR PIC18F4550

Este posee un periférico de comunicación USB, existen básicamente tres tipos de comunicación en protocolo USB:

- BULK USB
- HID USB
- CDC USB

Nosotros usamos el CDC USB que en términos mas explícitos seria el COMMUNICATIONS DEVICES CLASS, este tipo de comunicación nos perite realizar cualquier comunicación entre el microcontrolador y un programa de computadora (en nuestro caso JAVA).

Con la comunicación CDC USB podemos emular que la conexión es COM estándar, pero con la diferencia que se están enviando datos a mucha más velocidad, es decir cuando usamos la comunicación CDC es como si estuviéramos usando una comunicación serie normal.

Para esto debemos realizar un ajuste en el programa del microcontrolador para que la computadora lo reconozca como un puerto serie COM VIRTUAL, microchip nos provee de un driver para esta comunicación que es el **mchpcdc.inf**.

Que se instala de la siguiente forma:

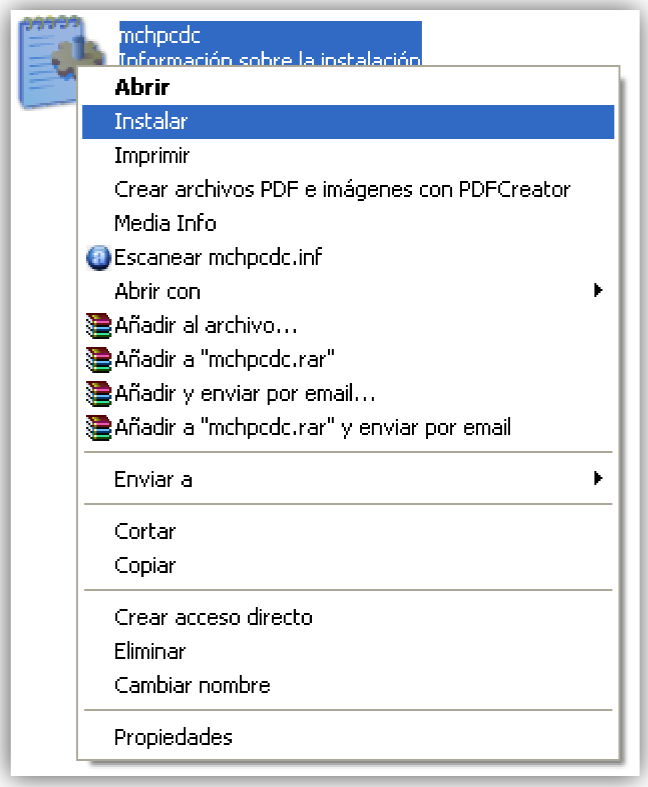

FIGURA Nº 2
Instalación del driver USB de microchip

Este proceso de instalación no produce ningún tipo de ventana se la realiza de forma inmediata, con este driver instalado y grabado el microcontrolador con un firmware incluya la gestión del puerto ya podemos realizar la comunicación entre el microcontrolador y la pc.

El firmware que debemos incluir en el programa del microcontrolador está incluido en el archivo .rar que acompaña a este proyecto y en el punto de programa microcontrolador se aprecia como se lo debe incluir.

2.2 CIRCUITO ELECTRONICO

El circuito electrónico usado para este proyecto, lo realizamos en el software PROTEUS que es muy usado para simulación de circuitos basados en microcontroladores, y en sus nuevas versiones incluye en conector USB para las simulaciones.

Circuito electrónico usado:

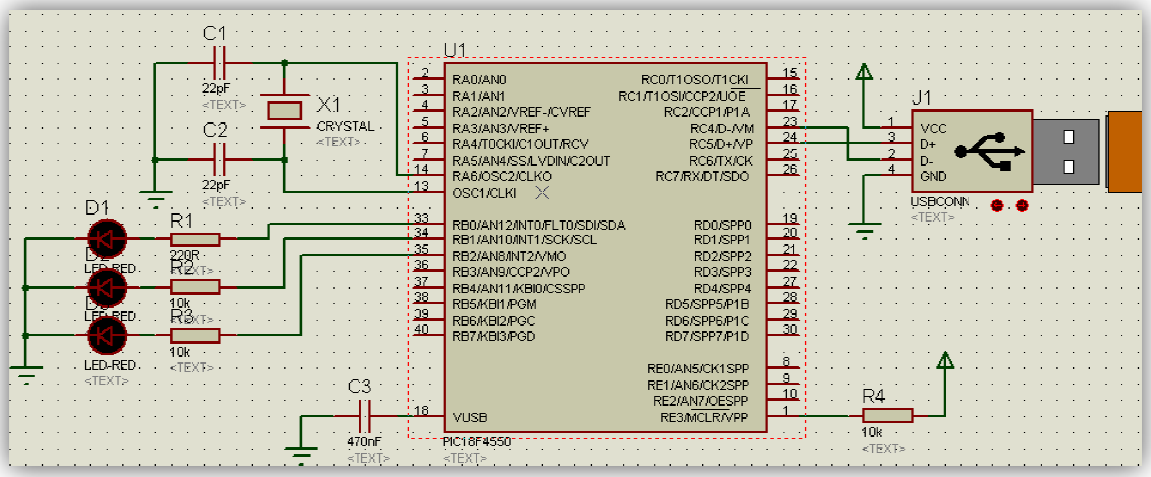

FIGURA Nº 3
Circuito Electrónico usado para el proyecto

Como se aprecia en esta ultima figura, se está conectando un circuito de reloj con un cristal resonador de 20 MHz, los leds simulan las lámparas de tres habitaciones, estos tres bits pueden ser ingresados en un circuito de potencia con TRIACS y encenderían lámparas de 100 W de potencia activa.

En el proyecto usaremos la tensión de 5V que entrega el puerto USB de la computadora, entrega una corriente de 500mA suficiente para encender los leds y alimentar al circuito eléctrico.

2.3 ECLIPSE

Usamos el programa eclipse para la compilación de nuestro proyecto, sobre este usaremos un driver que se encargara de abrir el puerto serie para este programa.

2.4 DRIVER PUERTO SERIE

El driver para puerto serie se puede descargar gratuitamente de internet, está en formato .rar y se puede buscar por el nombre GiovynetDriver.rar una vez descargado mostrara:

FIGURA Nº 4
Driver Giovynet para abrir el puerto serie con JAVA

Abrimos el programa eclipse y realizamos un nuevo proyecto:

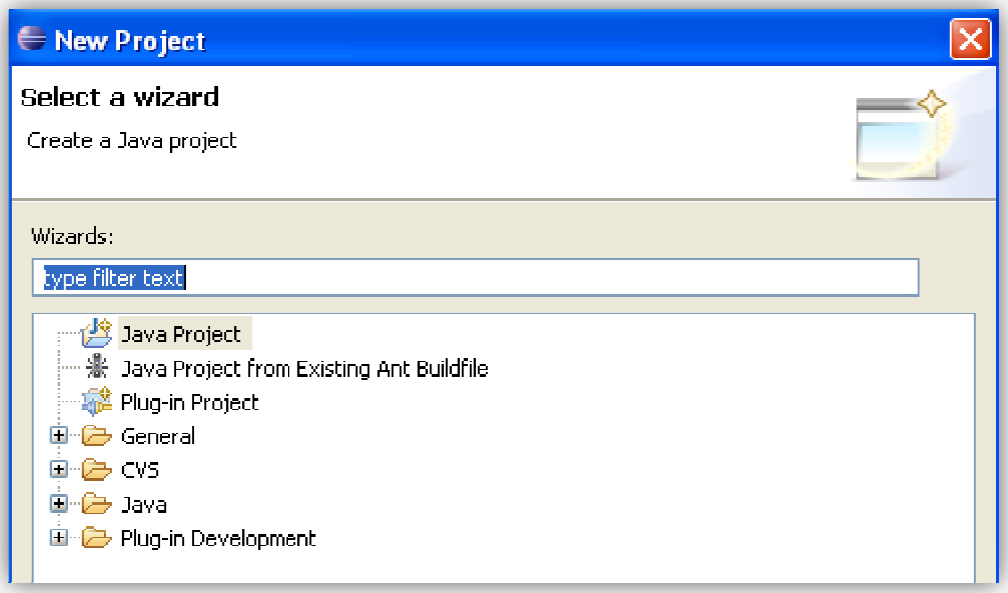

FIGURA Nº 5
Nuevo Proyecto JAVA

Escogemos en Java Project, aparecerá una nueva ventana y le damos como nombre RS232_10 y seleccionamos la opción crear proyecto desde una fuente existente:

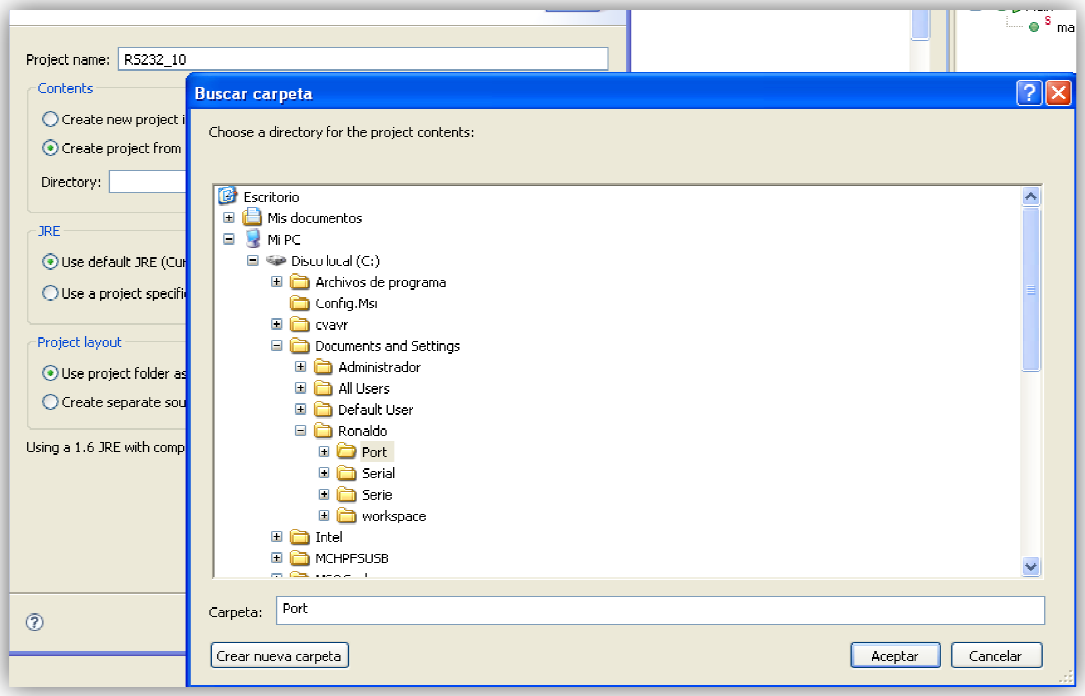

FIGURA Nº 6
Nuevo Proyecto JAVA

En el archivo .rar se incluye la carpeta serial dentro de la carpeta JAVA, seleccionamos la misma, en mi computadora se encuentra en el escritorio:

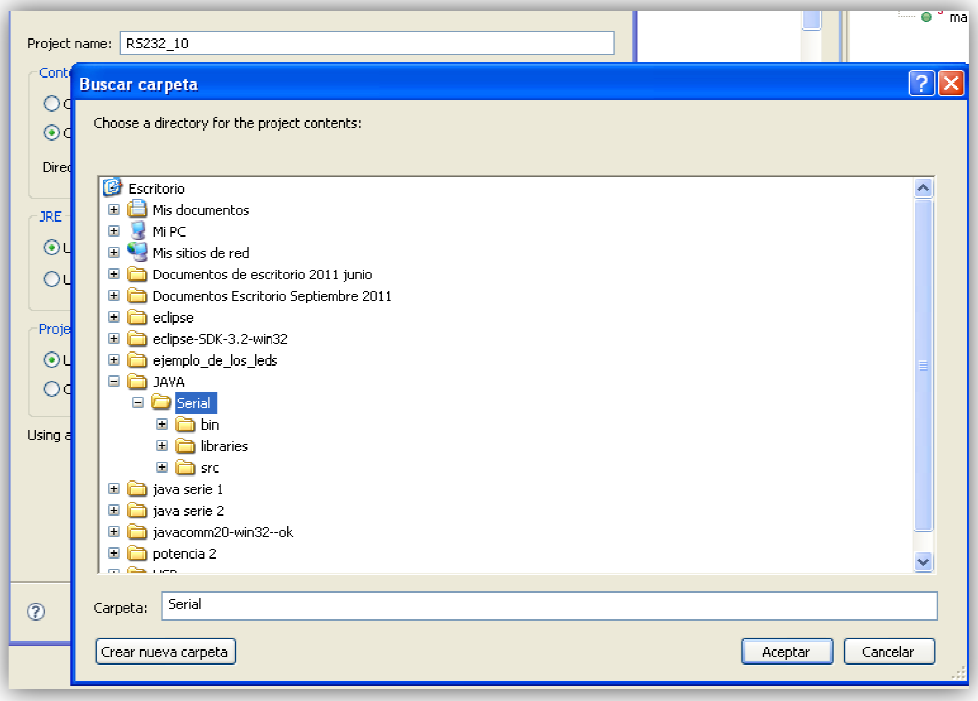

FIGURA Nº 7
Nuevo Proyecto JAVA de fuente existente

Clic en aceptar y finish deberíamos ver que cargo el programa que estaba dentro de la carpeta, el código que se ve en la figura debajo es un ejemplo que imprime **Hola Ramirex** (este programa es de ejemplo el que está en la carpeta es el que se utilizo en el programa final):

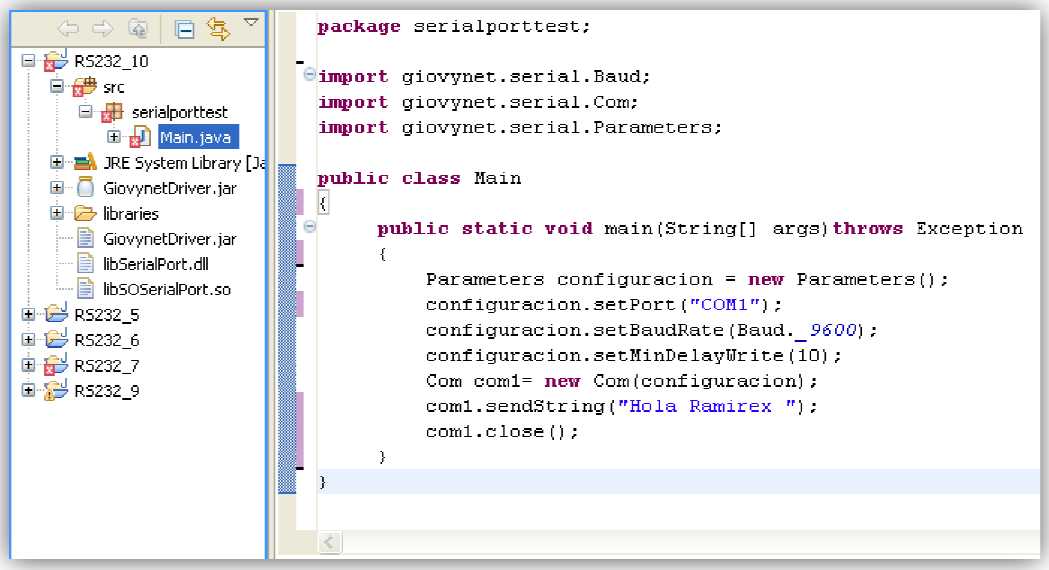

FIGURA Nº 7
Programa ejemplo del puerto serie

Para realizar una verificación que si el programa está enviando los datos correctamente usaremos el programa Virtual Serial Port que realiza la conexión virtual de dos puerto serie virtuales (no existen físicamente):

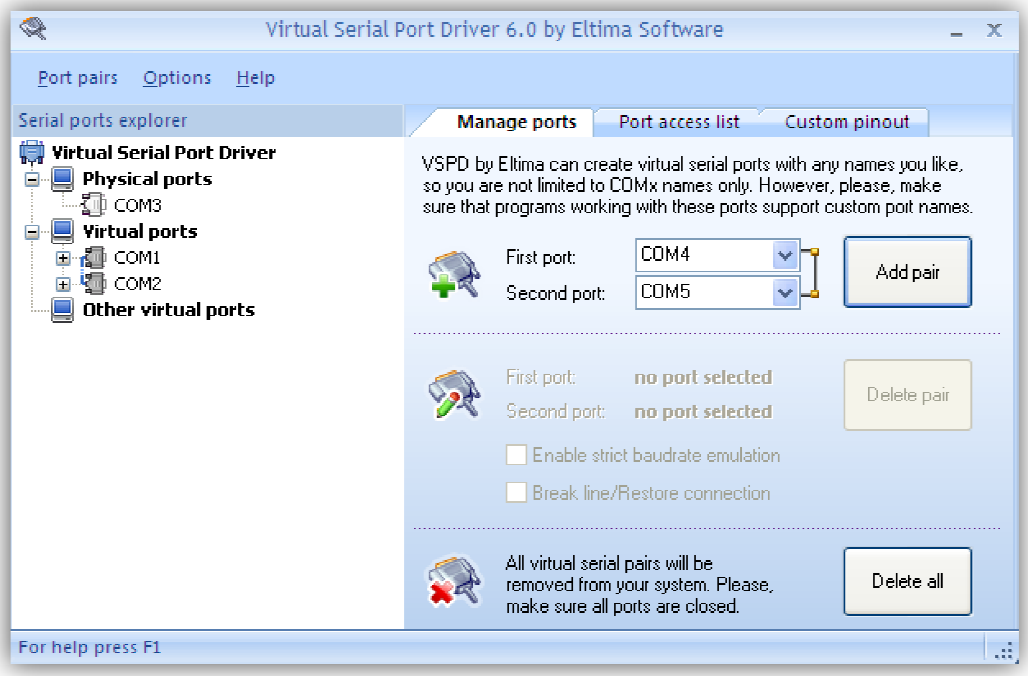

FIGURA Nº 8
Programa Virtual Serial Port

Una vez hecho esto abrimos el hiperterminal de Windows (si no se tiene se puede descargar de la red), con el que conectaremos nuestro programa JAVA, el programa en JAVA esta configurado en el COM1 y el hiperterminal debe estar conectado en el COM2 según la conexión que hicimos con el Virtual Serial Port.

FIGURA Nº 9
Nombre asignado a la comunicación con hiperterminal

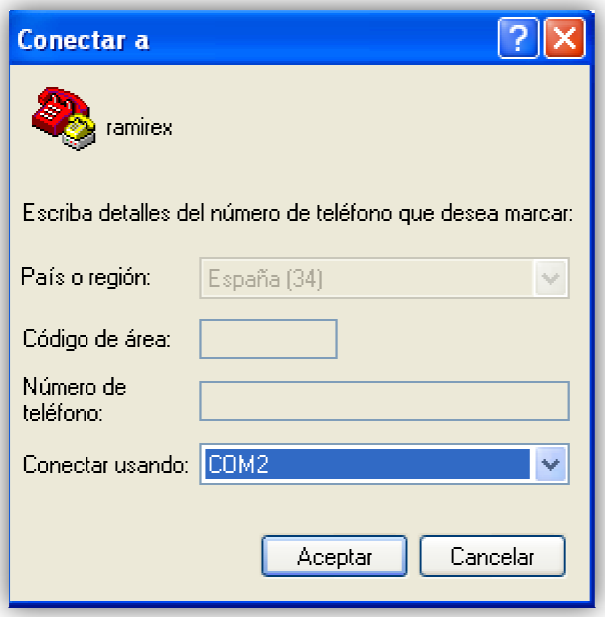

FIGURA Nº 10
Dirección al COM2 con el hiperterminal

Las propiedades de la comunicación son:

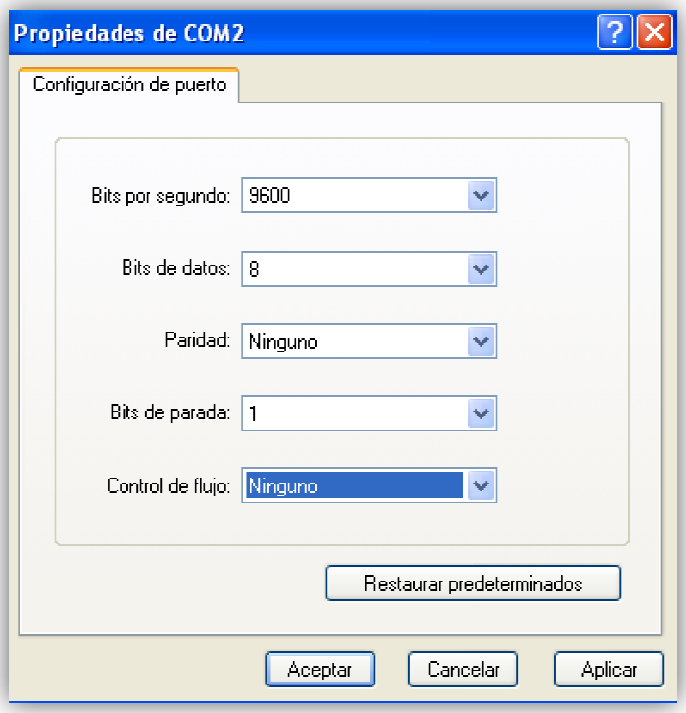

FIGURA Nº 11
Propiedades de comunicación del COM2

Clic en aceptar y el hiperterminal está listo para recibir datos por el puerto serie COM2, nos muestra la siguiente figura mostrando cono característica CONECTADO:

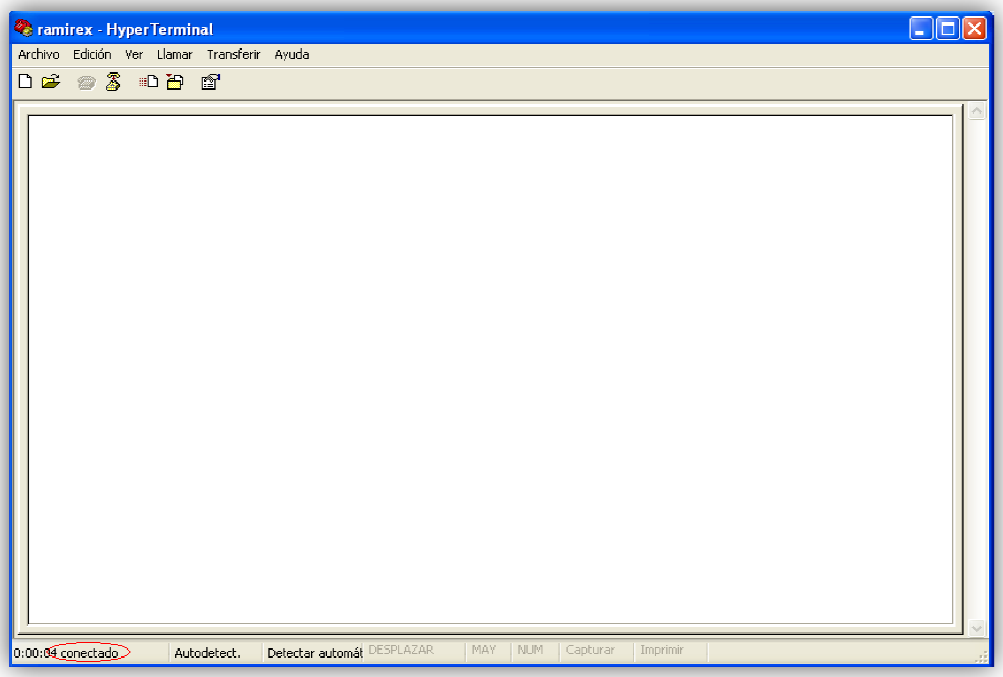

FIGURA Nº 12
Ventana hiperterminal conectada al COM2

En el programa java realizamos la ejecución del mismo como Java Aplication:

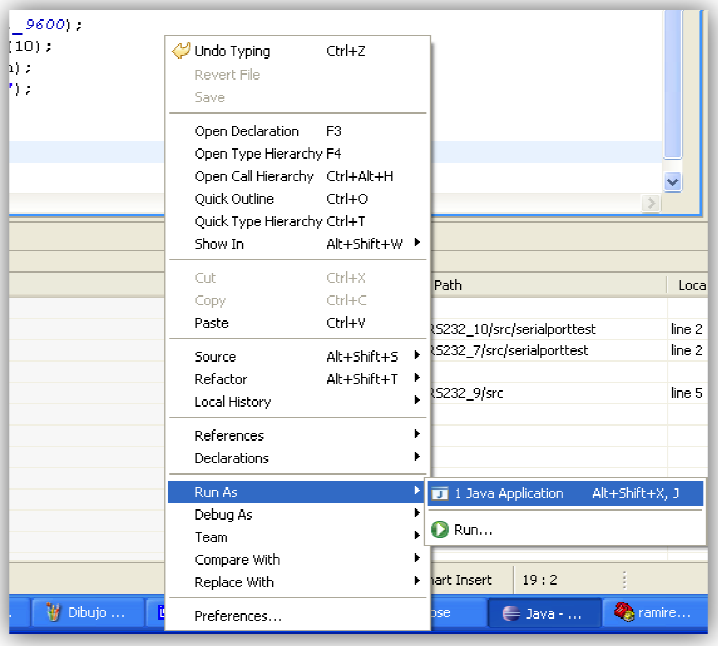

FIGURA Nº 13
Ejecución del programa en JAVA conectada al COM1

Y veremos en el hiperterminal:

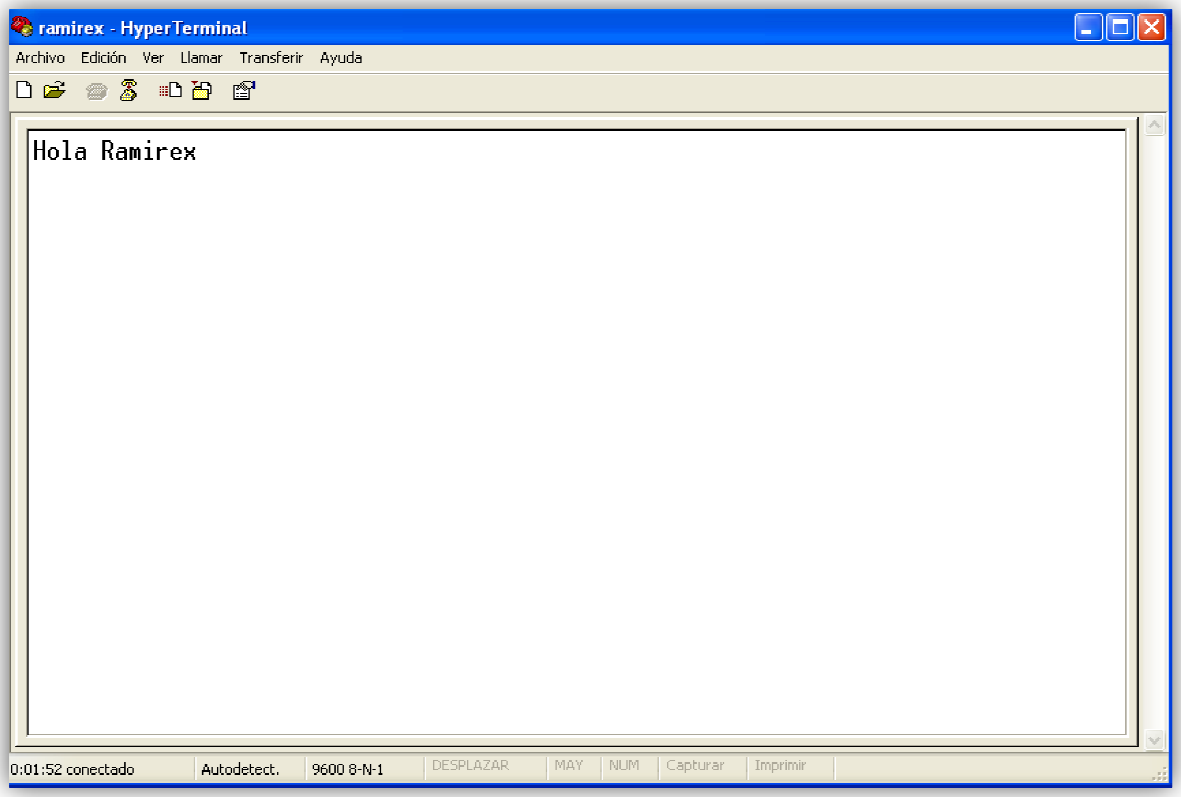

FIGURA Nº 14
Verificación de envío del COM1 al COM2

Este ejemplo imprime **Hola Ramirex**, ahora realizaremos un programa que envié datos numéricos para el proyecto seria enviar:

Dato 11	**Enciende la lámpara 1**
Dato 12	**Apaga la lámpara 1**
Dato 13	**Enciende la lámpara 2**
Dato 14	**Apaga la lámpara 2**
Dato 15	**Enciende la lámpara 3**
Dato 16	**Apaga la lámpara 3**

El programa correspondiente para este procedimiento es el que esta descrito en el punto programa java una vez ejecutado con el Run-Java Aplication se aprecia en la consola de Eclipse:

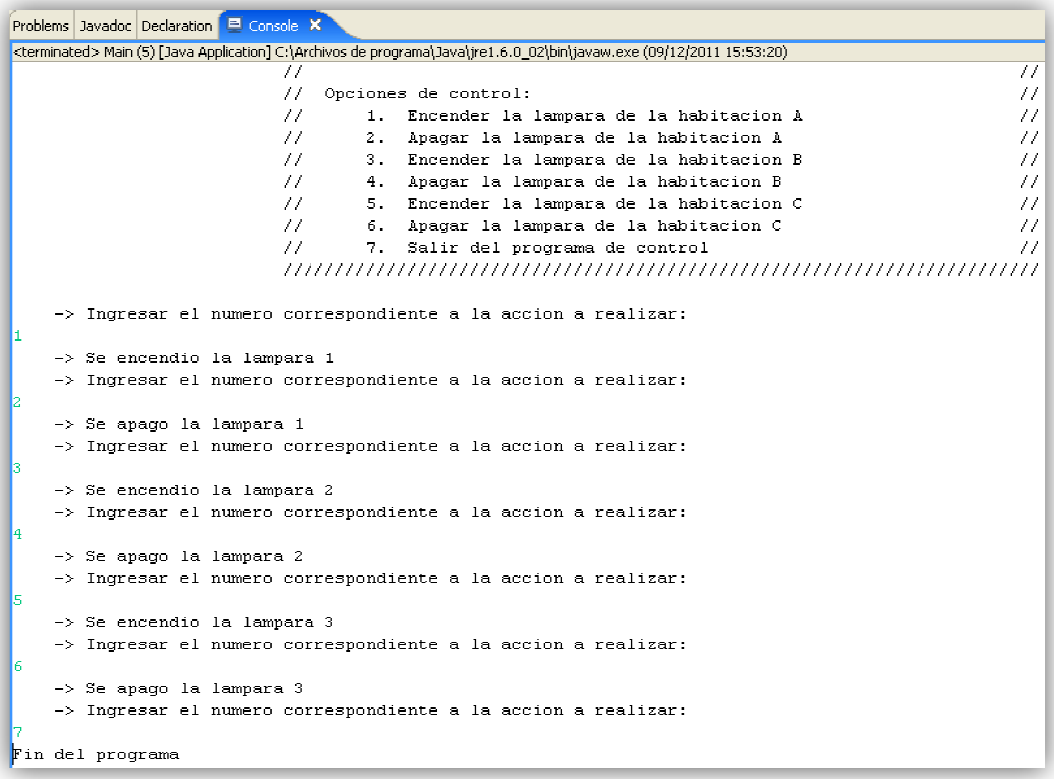

FIGURA Nº 15
Pestaña consola del programa JAVA en Eclipse

Apreciándose los datos en el hiperterminal los datos 11, 12, 13, 14, 15 y 16 imprimidos lado a lado:

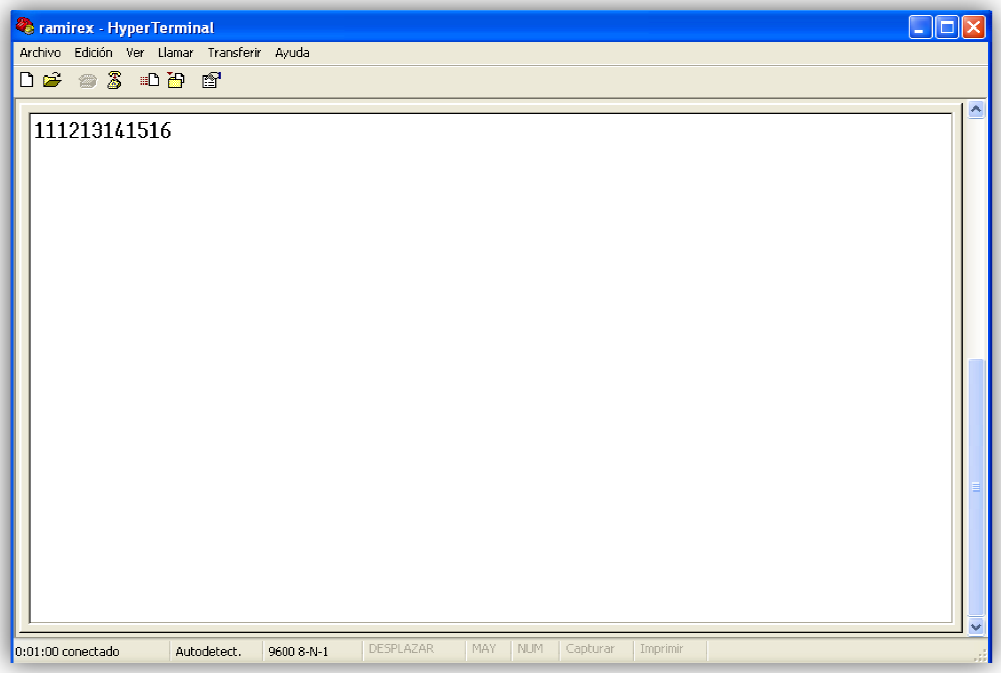

FIGURA Nº 15
Verificación de envió de datos al COM2

Ahora nos toca con el microcontrolador.

3. PROGRAMA MICROCONTROLADOR

El programa del microcontrolador fue realizado en CCS PCHW, que realiza la apertura del puerto USB en CDC y captura los datos que envía el programa en JAVA y entra en un menú de selección en el cual enciende o apaga los leds, que en nuestro caso simulan el circuito de iluminación de una vivienda.

El programa es el siguiente:

```
#include <18F4550.h>
#fuses HSPLL,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL5,CPUDIV1,VREGEN
#use delay(clock=48000000)
#USE fast_IO (B)
#include ".\incluir\usb_cdc_ramirex.h"

void main() {

  BYTE value;
  delay_ms(300);
  SET_TRIS_B(0x00);
  OUTPUT_B(0);

  usb_cdc_init();
  usb_init();

  do
  {
    value = gethex_usb();

    switch(value)
      {
      case 0x11:      output_high(PIN_B0);
                      delay_ms(100);
                      break;
      case 0x12:      output_low(PIN_B0);
                      delay_ms(100);
                      break;
      case 0x13:      output_high(PIN_B1);
                      delay_ms(100);
                      break;
      case 0x14:      output_low(PIN_B1);
                      delay_ms(100);
                      break;
```

```
      case 0x15:      output_high(PIN_B2);
                      delay_ms(100);
                      break;
      case 0x16:      output_low(PIN_B2);
                      delay_ms(100);
                      break;
      default:        output_high(PIN_B0);
                      output_high(PIN_B1);
                      delay_ms(50);
      }
    delay_ms(100);
  } while (TRUE);
}
```

El firmware que se carga es él **".\incluir\usb_cdc_ramirex.h"**, que también les incluyo en el archivo comprimido.

Este programa se debe cargar al chip usando un programador de microcontroladores, el archivo que se debe cargar en el microcontrolador es el .hex

Cuando se conecta el microcontrolador con la computadora por primera vez se aprecia:

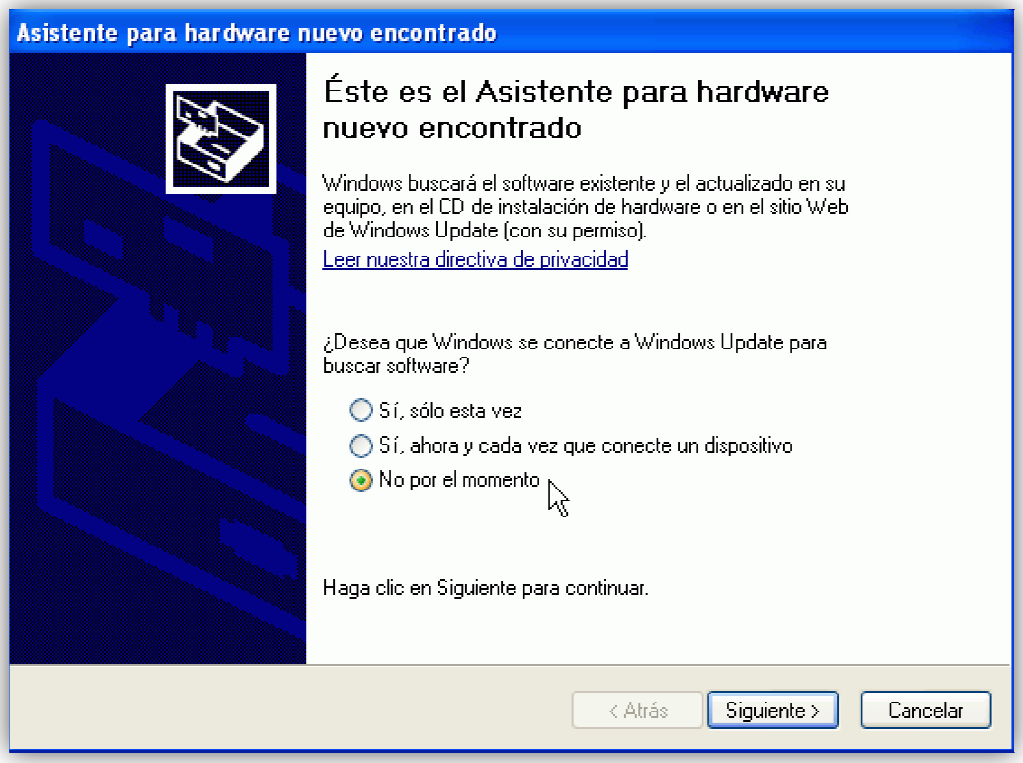

FIGURA Nº 16
Configuración de nuevo hardware

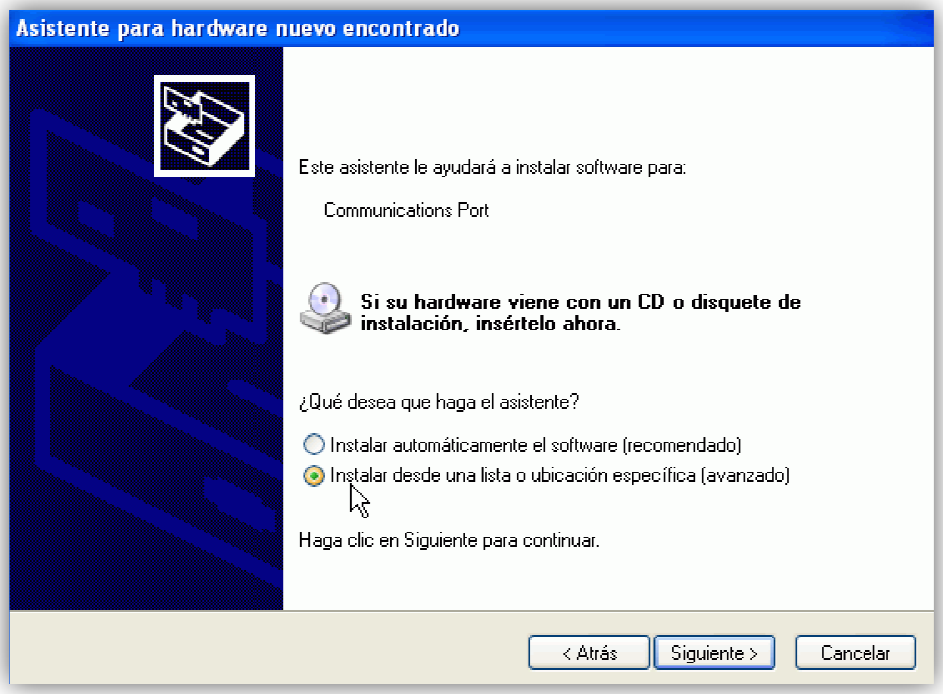

FIGURA Nº 17
Configuración de nuevo hardware

Dar la ruta a la carpeta driver del archivo comprimido.

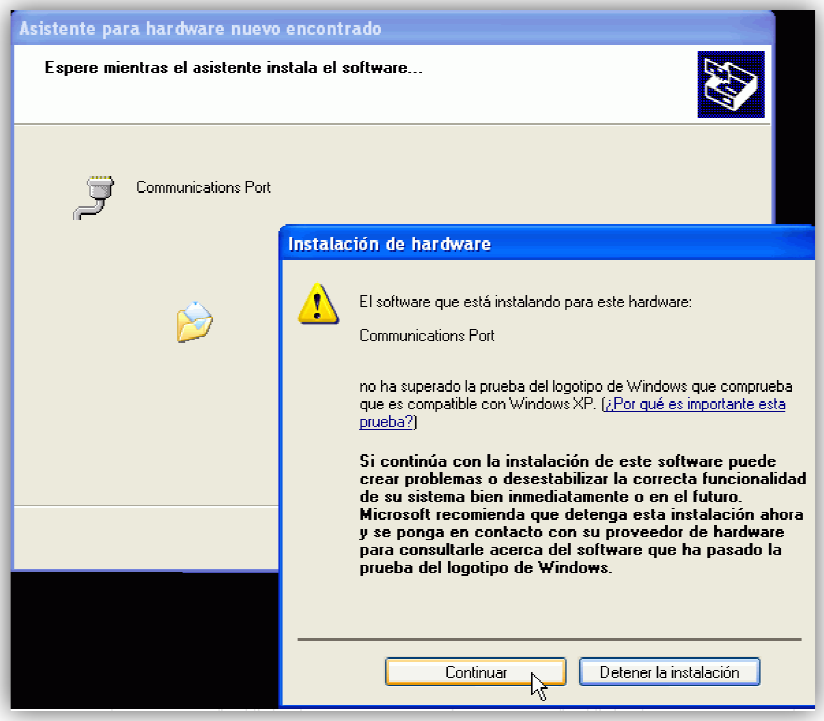

FIGURA Nº 17
Configuración de nuevo hardware

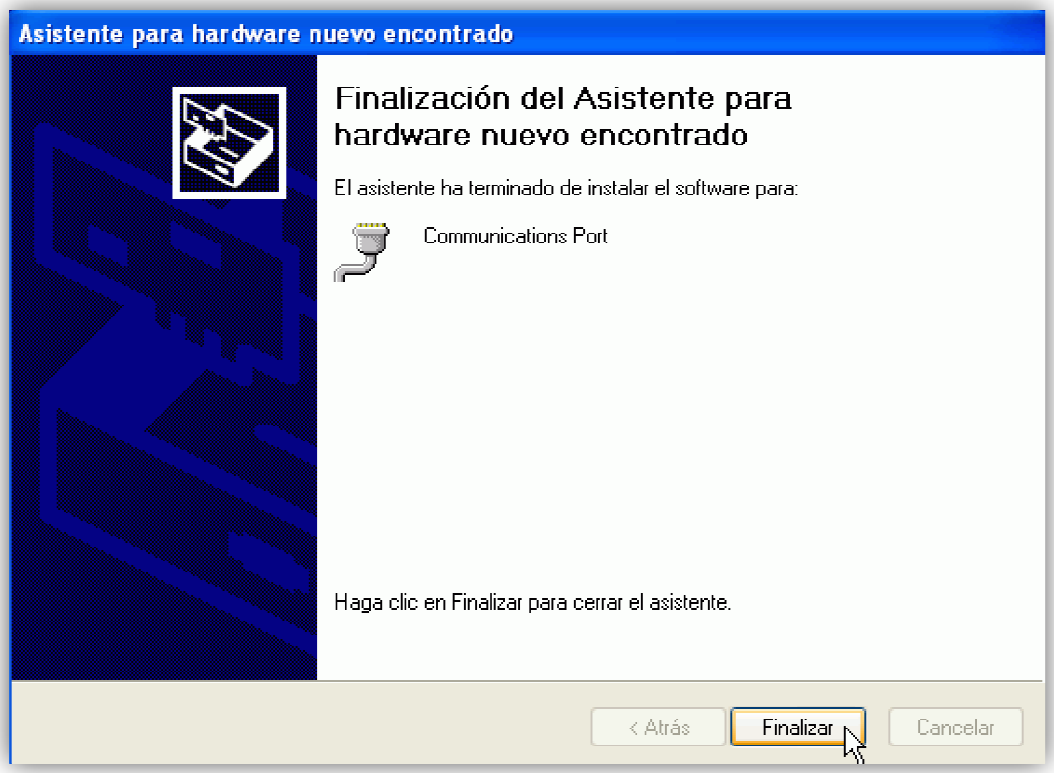

FIGURA Nº 18
Finalización de configuración

Para verificar la conexión del nuevo hardware seguimos los siguientes pasos:

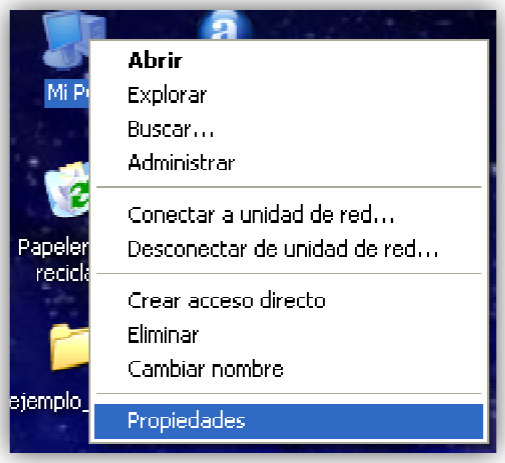

FIGURA Nº 19
Propiedades de PC

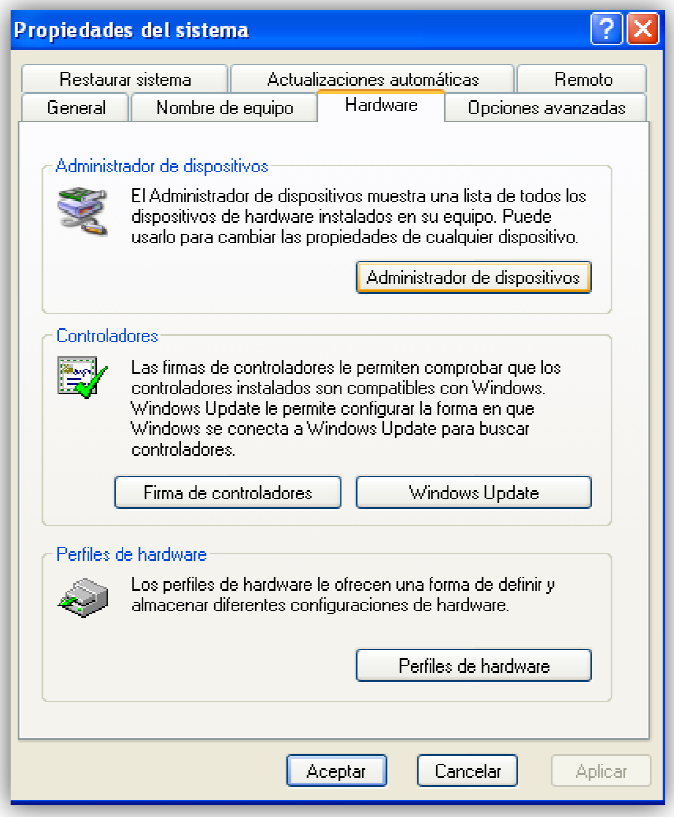

FIGURA Nº 20
Propiedades del sistema

Clic en Administrador de dispositivos:

FIGURA Nº 21
Administrador de dispositivos

En la figura 21 se aprecia al nuevo puerto COM19, por el cual nos comunicaremos entre el microcontrolador y el programa JAVA, también podemos ver las propiedades del COM19.

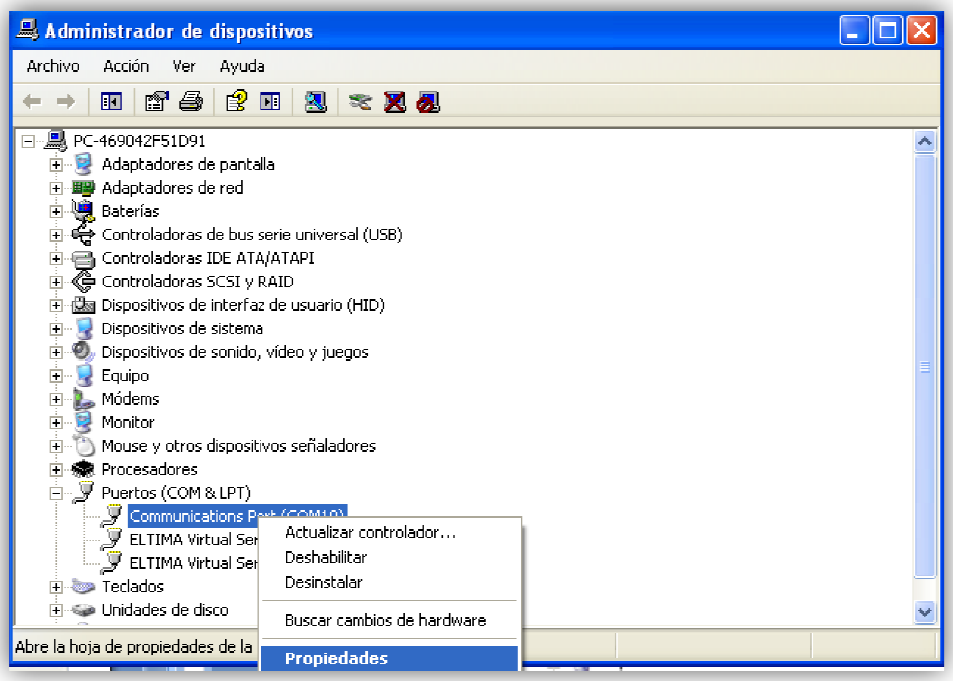

FIGURA Nº 22
Propiedades del COM19

FIGURA Nº 23
Propiedades del COM19

Ahora el microcontrolador está listo para enviar y recibir datos por el puerto USB en CDC, cabe mencionar como se aprecia en la última grafica el puerto esta denominado como COM19 entonces el programa en JAVA también debe estar configurado en COM19, esta denominación no es constante y puede variar según el equipo.

4. PROGRAMA JAVA

```
package serialporttest;


import java.io.BufferedReader;
import java.io.InputStreamReader;

import giovynet.serial.Baud;
import giovynet.serial.Com;
import giovynet.serial.Parameters;

public class Main

{
    public static void main(String args[]) throws Exception
    {
      Parameters configuracion = new Parameters();
          configuracion.setPort("COM19");
          configuracion.setBaudRate(Baud._9600);
          configuracion.setMinDelayWrite(10);
          Com com1= new Com(configuracion);

      BufferedReader in;
        in = new BufferedReader(new InputStreamReader(System.in));
        int m = 0;


System.out.println (" ");
System.out.println ("////////////////////////////////////////////////");
System.out.println ("//     PROGRAMACION PARA INGENIERIA II           //");
System.out.println ("//      PROYECTO DE FINAL DE SEMESTRE            //");
System.out.println ("//  CONTROL DE LAMPARAS A TRAVES DEL PUERTO USB   //");
System.out.println ("//                                               //");
System.out.println ("//  Opciones de control:                         //");
System.out.println ("//  1.  Encender la lampara de la habitacion A   //");
System.out.println ("//  2.  Apagar la lampara de la habitacion A     //");
System.out.println ("//  3.  Encender la lampara de la habitacion B   //");
System.out.println ("//  4.  Apagar la lampara de la habitacion B     //");
System.out.println ("//  5.  Encender la lampara de la habitacion C   //");
System.out.println ("//  6.  Apagar la lampara de la habitacion C     //");
System.out.println ("//  7.  Salir del programa                       //");
System.out.println ("////////////////////////////////////////////////");


        System.out.println (" ");

        do
          {
System.out.println ("-> Ingresar el numero correspondiente a la acción a
realizar:");
        m = Integer.parseInt (in.readLine ());

        switch (m)
```

```
            {
                 case 1:
                     System.out.println ("    -> Se encendio la lampara 1");
                     com1.sendString("11");
                     break;
                 case 2:
                     System.out.println ("    -> Se apago la lampara 1");
                     com1.sendString("12");
                     break;
                 case 3:
                     System.out.println ("    -> Se encendio la lampara 2");
                     com1.sendString("13");
                     break;
                 case 4:
                     System.out.println ("    -> Se apago la lampara 2");
                     com1.sendString("14");
                     break;
                 case 5:
                     System.out.println ("    -> Se encendio la lampara 3");
                     com1.sendString("15");
                     break;
                 case 6:
                     System.out.println ("    -> Se apago la lampara 3");
                     com1.sendString("16");
                     break;
                 case 7:
                     System.out.println ("Fin del programa \n");
                     com1.close();
                     break;
               }
            }
            while (m != 7);
        }
}
```

Hacer correr el programa con clic derecho sobre algún espacio vacio del código Run-Java Aplication y según se escoja el menú de opciones de enecederan o apagaran los leds.

Para una apreciación grafica se puede ver el funcionamiento del mismo en **YOUTUBE** con el nombre **PUERTO USB CON JAVA UMSA INGENIERIA**

5. BIBLIOGRAFIA

Las páginas usadas que nos ayudaron y merecen el reconocimiento nuestro son:

- La pagina se sixca, con ejemplos de aplicación USB
- MuchoTrasto – ejemplos de aplicaciones USB CDC
- La pagina de Red-Raven con explicación adicional y muy buen contenido.
- No podía faltar la pagina de www.todopic.com
- Libro Compilador C CCS y Simulador Proteus – Eduardo Garcia Brejio
- La biblia de Java
- Aprenda Java como si estuviera en primero