PROYECTO: ASISTENTE PARA AFONIA FUNCIONAL

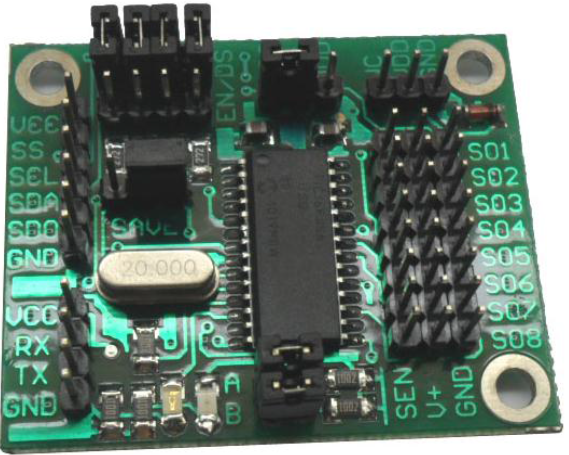
 |

PROYECTO DE TESIS

2014

**IMPLEMENTACION HARWARE**

1. **MATERIALES UTILIZADOS**
   1. **TARJETA DE CONTROL CON PIC 16F886**

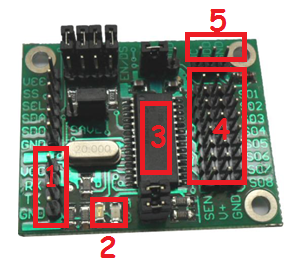


ESPECIFICACIONES

* Comunicación por UART.
* 10 entradas y salidas digitales.
* 1 led indicador de presencia de voltaje.
* Programación por ICSP.
* Fuente de voltaje mínima 4.5 v
* Fuente de voltaje máxima 5.0 v

CONECTOR DE PROGRAMACION

DIAGRAMA DE PINES



1. Conector con antenas de comunicación inalambrica Xbee y Bluetooth
2. Led de encendido y alimentación de voltaje DC.
3. Microcontrolador 16f886.
4. Entradas y salidas digitales.
5. Selector de alimentación de voltaje externo.
   1. **ANTENA DE BLUETOOTH HC06**



La antena de Bluetooth HC06 transmite los caracteres escritos en la interfaz en java hacia el teléfono por el terminal de conexión TX del micontrolador.

Este dispositivo Bluetooth es de clase 1. Denotado por su largo alcance de hasta 100 metros. Consume 120mA y tiempo de uso del dispositivo menor a 6 horas.

La tasa de transmisión de bits está configurada desde fábrica a 9600 baudios y es reconfigurable por comandos AT.

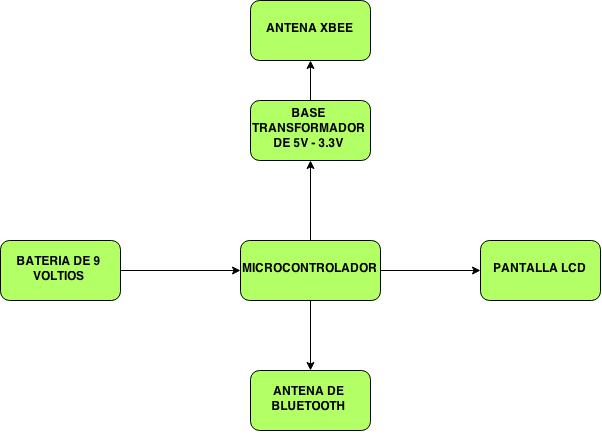
* 1. **ANTENA XBEE**



La interfaz gráfica desarrollada en java permite digitar caracteres desde el teclado del computador y conectado a una antena Xbee, transmiten sus datos hacia el concentrador de datos para ser mostrados en el LDC 16x2 y retransmitidos a la aplicación instalada en el teléfono y ser reproducidos .

* Se alimenta de 5 voltios DC, la señal de control son digitales.
* Consume de 20mW.
* La tasa de trasmisión de bits (BAUDRATE) es de 9600.

1. **DIAGRAMA ESQUEMÁTICO CIRCUITO MASTER.**

****

1. **TABLA DE INTERCONEXION CIRCUITO MASTER.**

|  |  |
| --- | --- |
| LCD 16X2 | PIC 16F886 |
| RS | PIN RB0 |
| EN | PIN RB2 |
| D4 | PIN RA0 |
| D5 | PIN RA1 |
| D6 | PIN RA2 |
| D7 | PIN RA3 |
| - | GND |
| + | 5V |

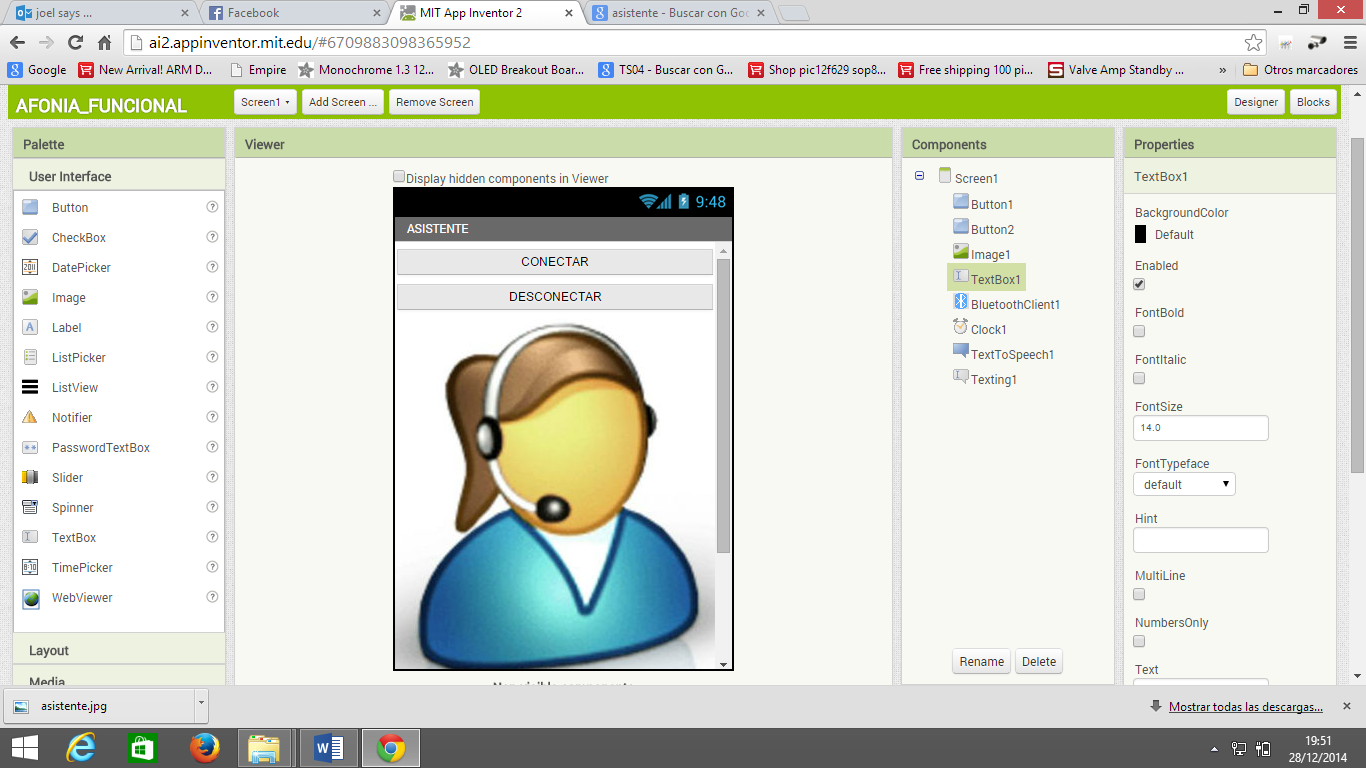
|  |  |
| --- | --- |
| ANTENA XBEE | PIC 16F886 |
| - | GND |
| + | + 5V |
| PIN TX | PIN RX |

|  |  |
| --- | --- |
| ANTENA BLUETOOTH | PIC 16F886 |
| - | GND |
| + | + 5V |
| PIN RX | PIN TX |

1. **APLICACIÓN ANDROID**

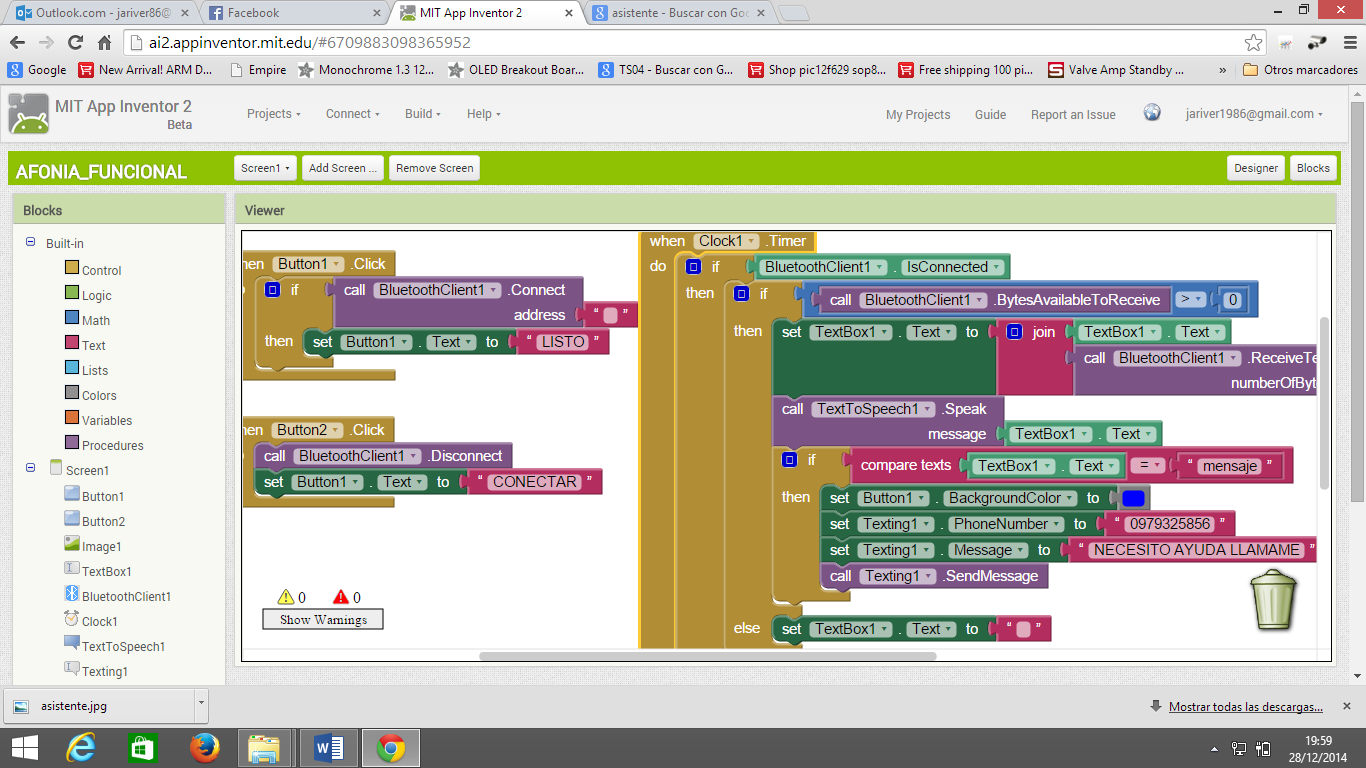
La interfaz presenta de desarrollo de la aplicación se compone de botoneras, servidor de comunicación bluetooth, reloj y convertidor de texto-audio.

1. **INTERFAZ PRINCIPAL.**



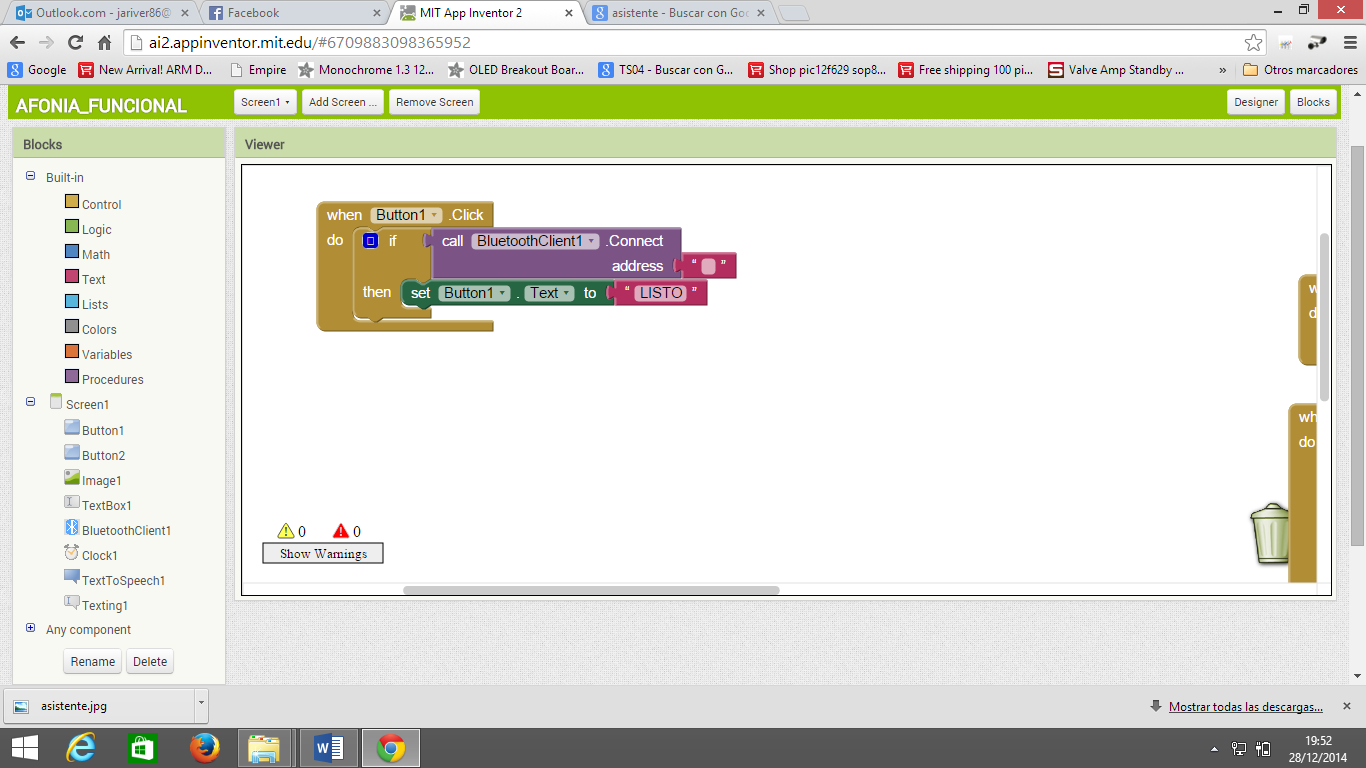
La aplicación es amigable con el usuario se necesita de presionar el botón de CONECTAR para que el teléfono empiece a recibir los datos enviados desde el microcontrolador concentrados de caracteres.

1. **INTERFAZ DE PROGRAMACIÓN**



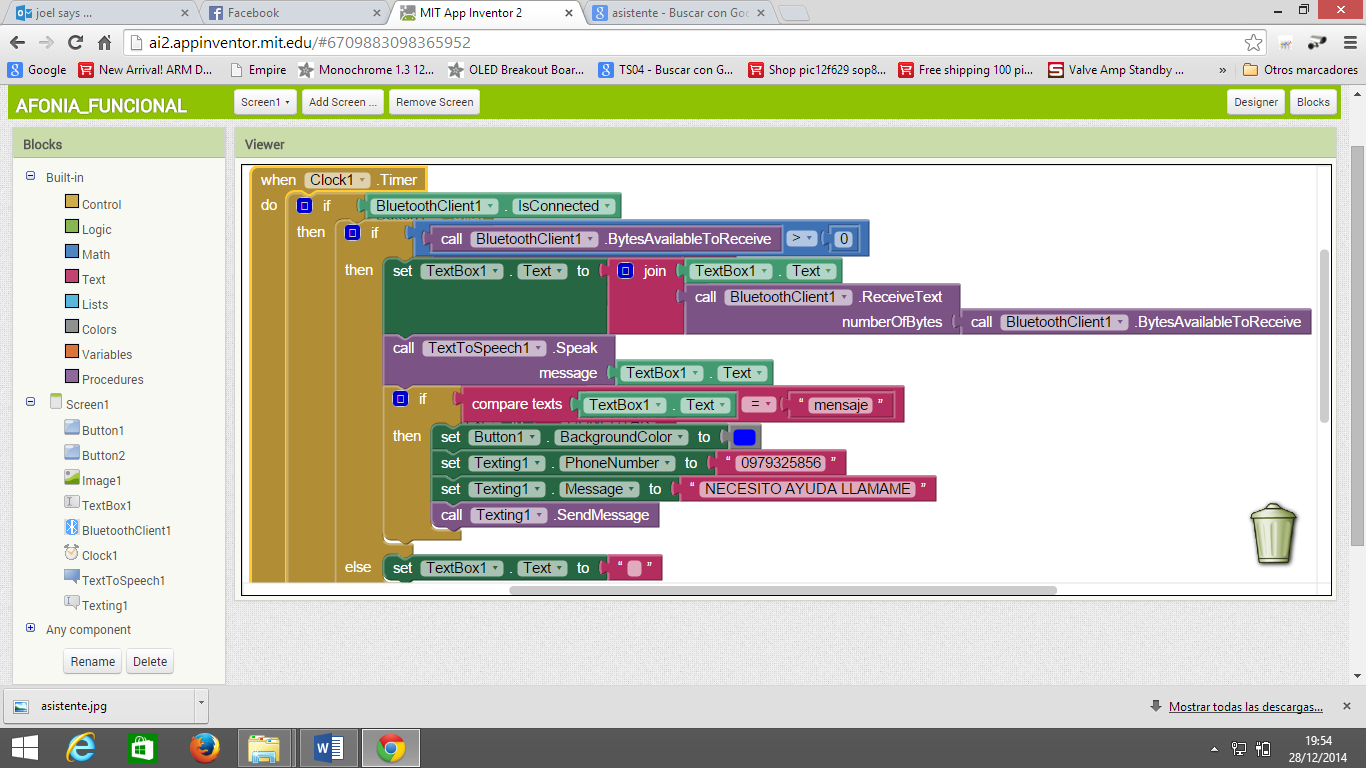
APP-Inventor presenta un block de diseño y hace uso de los elementos incorporados en el panel de componentes. Se utilizan sentencias de validación IF ELSE, WHILE, etc. Para la codificación de la recepción de datos vía bluetooth y la reproducción de os caracteres escritos por el usuario.

1. **CODIGO DE COMUNICACIÓN CON DISPOSITIVO.**



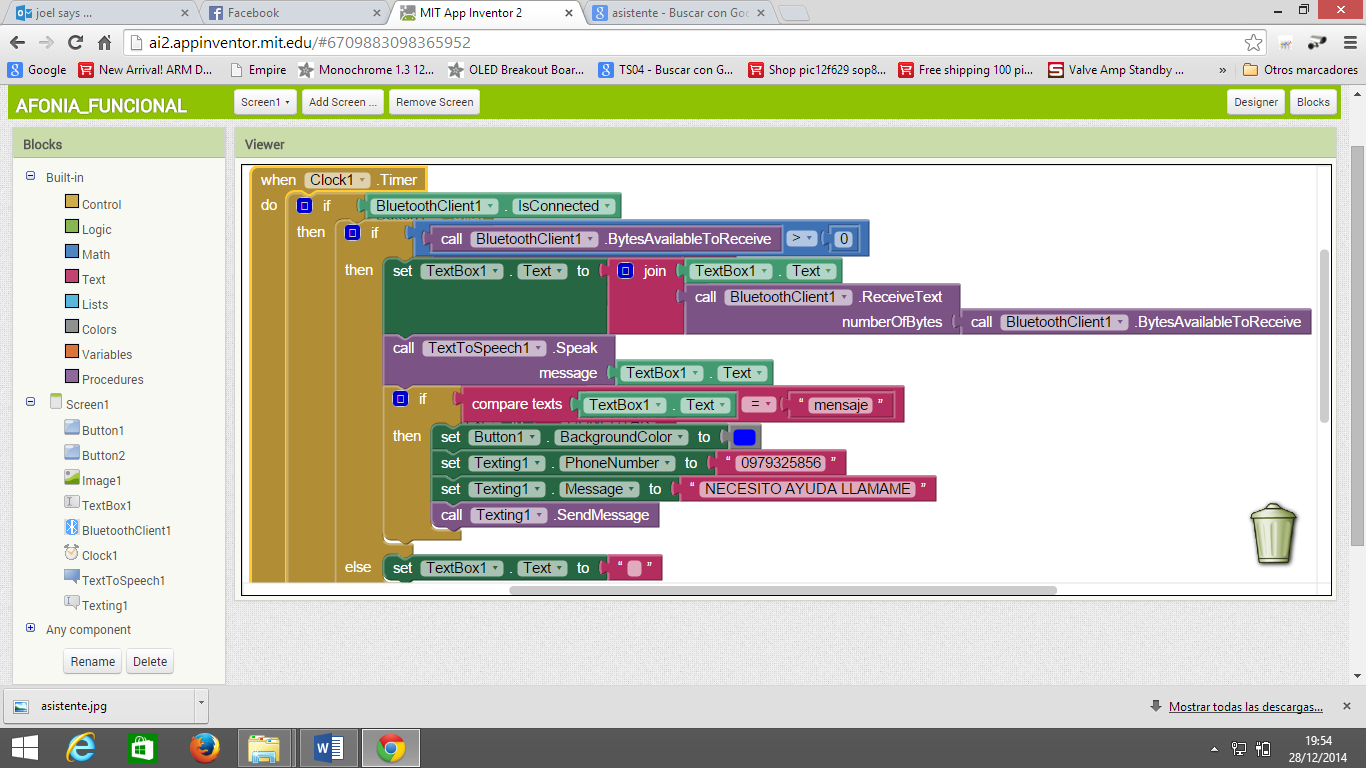
La presente interfaz presenta la codificación para el enlace de datos vía bluetooth, los caracteres recibidos son recibidos a frecuencia de la banda de 2,4 GHz con estándar ISM (Industrial, científica y médica).

1. **CODIGO DE RECEPCION DE DATOS.**



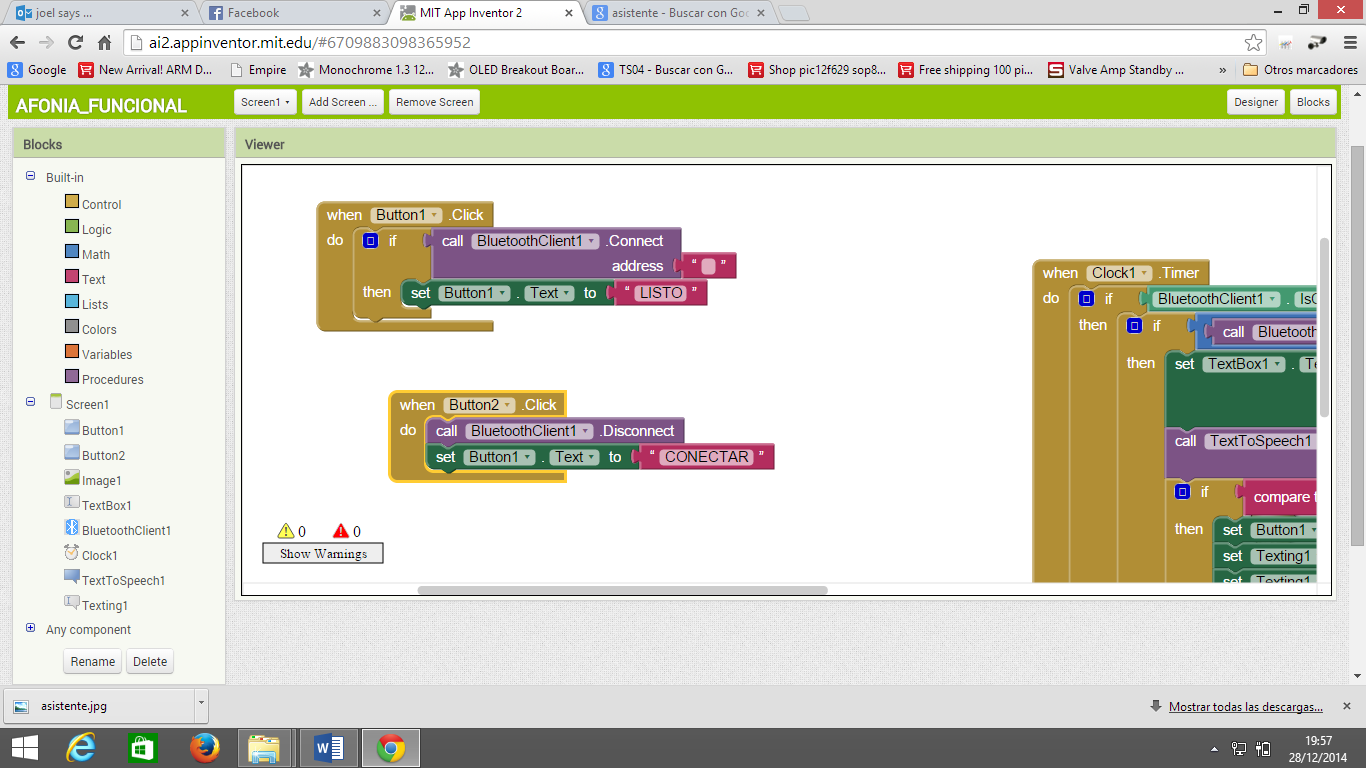
Los caracteres son almacenados en una variable y permiten interpretar con el fin de que el oyente entienda los pensamientos del usuario de este dispositivo.

1. **CODIGO DE REPRODUCCION DE TEXTO RECIBIDO.**



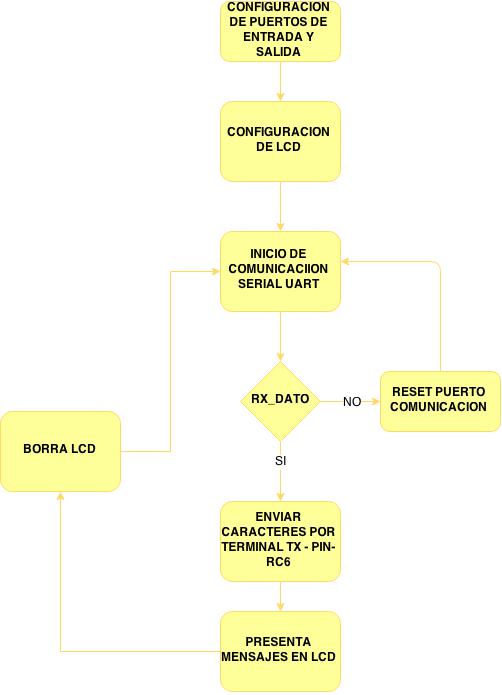
La función TextToSpeech reproduce en voz los caracteres digitados por el minusválido.

1. **CODIGO DE ROMPER ENLACE DE DATOS**



La función Disconnect rompe el enlace entre el teléfono y el dispositivo asistente.

1. **DIAGRAMA ASM FUNCIONAL**

****

1. **CODIGO DE PROGRAMACION MICROCONTROLADOR.**

**ASIGNACION DE LCD**

dim LCD\_RS as sbit at RB0\_bit

LCD\_EN as sbit at RB2\_bit

LCD\_D4 as sbit at RA0\_bit

LCD\_D5 as sbit at RA1\_bit

LCD\_D6 as sbit at RA2\_bit

LCD\_D7 as sbit at RA3\_bit

LCD\_RS\_Direction as sbit at TRISB0\_bit

LCD\_EN\_Direction as sbit at TRISB2\_bit

LCD\_D4\_Direction as sbit at TRISA0\_bit

LCD\_D5\_Direction as sbit at TRISA1\_bit

LCD\_D6\_Direction as sbit at TRISA2\_bit

LCD\_D7\_Direction as sbit at TRISA3\_bit

**CONFIGURACION DE TASA DE BITS**

UART1\_Init(9600)

**CONFIGURACION MENSAJE A LCD**

Lcd\_Init()

Lcd\_Cmd(\_LCD\_CLEAR)

Lcd\_Out(1,1,"AFONIA FUNCIONAL") ' escribe el mensaje en la primera fila

delay\_ms(2000) ' tiempo de espera

Lcd\_Cmd(\_LCD\_CLEAR) ' encera lcd

Lcd\_Cmd(\_LCD\_CURSOR\_OFF) ' apaga cursor

if (UART1\_Data\_Ready() <> 0) then ' Si el dato es recibido

UART1\_Read\_Text(TEXTO,":",32) ' lee cadenade caracteres

UART1\_Write\_Text(TEXTO) ' envia un eco de la cadena

Lcd\_Cmd(\_LCD\_CLEAR) ' encera lcd

Lcd\_Out(1,1,TEXTO) ' me muestra la cadena que escribiste via serial en el lcd

end if

1. **CODIGO DE PROGRAMACION GENERAL**

program afonia\_funcional

dim LCD\_RS as sbit at RB0\_bit

LCD\_EN as sbit at RB2\_bit

LCD\_D4 as sbit at RA0\_bit

LCD\_D5 as sbit at RA1\_bit

LCD\_D6 as sbit at RA2\_bit

LCD\_D7 as sbit at RA3\_bit

LCD\_RS\_Direction as sbit at TRISB0\_bit

LCD\_EN\_Direction as sbit at TRISB2\_bit

LCD\_D4\_Direction as sbit at TRISA0\_bit

LCD\_D5\_Direction as sbit at TRISA1\_bit

LCD\_D6\_Direction as sbit at TRISA2\_bit

LCD\_D7\_Direction as sbit at TRISA3\_bit

dim TEXTO as char[33]

main:

PORTA = %00000000

PORTB = %00000000

PORTC = %00000000

TRISA = %00000000

TRISB = %00000000

TRISC = %10000000

ANSEL = %00000000

ANSELH= %00000000

UART1\_Init(9600)

Lcd\_Init()

Lcd\_Cmd(\_LCD\_CLEAR)

Lcd\_Out(1,1,"AFONIA FUNCIONAL") ' escribe el mensaje en la primera fila

Lcd\_Out(2,1,"POR: IDETEC") ' escribe el mensaje en la segunda fila

delay\_ms(2000) ' tiempo de espera

Lcd\_Cmd(\_LCD\_CLEAR) ' encera lcd

Lcd\_Cmd(\_LCD\_CURSOR\_OFF) ' apaga cursor

while 1

if (UART1\_Data\_Ready() <> 0) then ' Si el dato es recibido

UART1\_Read\_Text(TEXTO,":",32) ' lee la cadena hasta que encuentre la letra K mayuscula

UART1\_Write\_Text(TEXTO) ' envia un eco de la cadena

Lcd\_Cmd(\_LCD\_CLEAR) ' encera lcd

Lcd\_Out(1,1,TEXTO) ' me muestra la cadena que escribiste via serial en el lcd

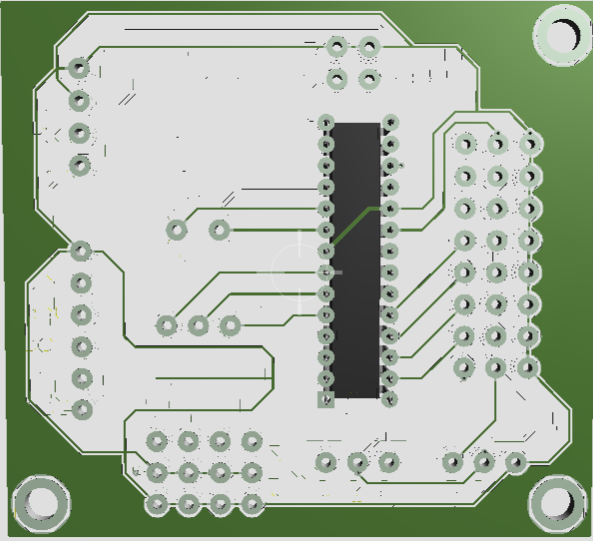
Delay\_ms(1000)

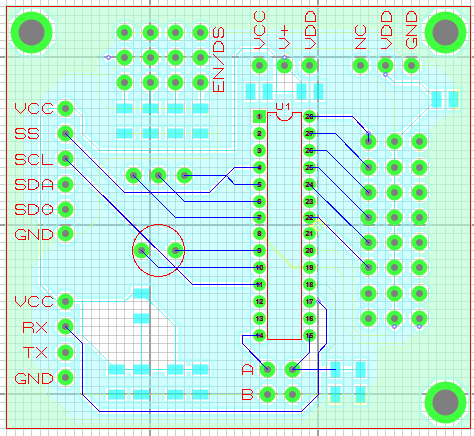
end if

wend

end.

1. DISEÑO DE PCB EN PROTEUS





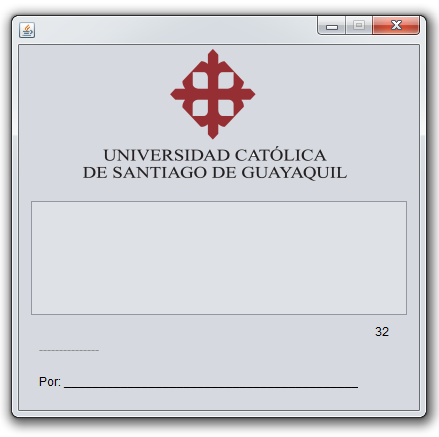
**IMPLEMENTACION SOFTWARE EN JAVA**

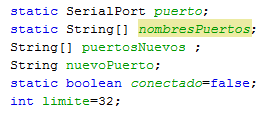
En Este proyecto se usara la librería JSSC (Java-Simple-Serial-Connector).

JSSC es una librería para java que nos permite comunicarnos con puertos Seriales, y es soportado en varios Sistemas Operativos, Para mas información y Consultas acerca de JSSC , visitar,

https://code.google.com/p/java-simple-serial-connector/

La interfaz del programa es la siguiente:

****

**Declaración de Variables**

Esas son las variables que se usaran donde:

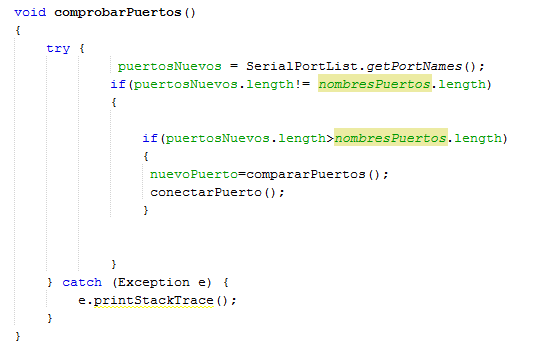
Puerto: es el nombre de la variable con el puerto COM que haremos la comunicaciones, y mantener la conexión

NombresPuertos: es un arreglo donde almacenaremos temporalmente el nombre de todos los puertos COM conectados a la pc.

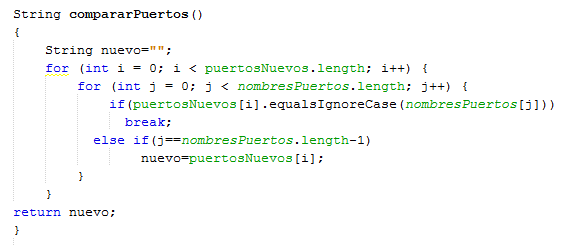
PuertosNuevos: es un arreglo donde almacenaremos temporalmente el nombre de los puertos nuevos que sean detectados

Conectado: es una bandera para saber el estado de la conexión

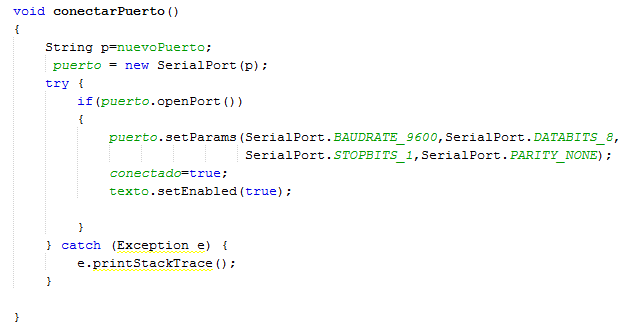
Límite: es una variable tipo entero para señalar el límite de caracteres que se podrá escribir en la pantalla

**Método comprobarPuertos**

Este Método estará corriendo en segundo plano para detectar nuevas conexiones, estará en un hilo, antes de iniciar la conexión, para así poder detectar el dispositivo para hacer la comunicación Serial , se estará actualizando el arreglo puertosNuevos llamando al método getPortNames , que es una función que nos provee la librería JSSC para obtener la lista de los puertos , entonces si la lista de puertos Nuevos es diferente a la lista de los puertos Listados anteriormente, entonces asignamos a la variable nuevoPuerto el nombre del puerto que nos retornara el método compararPuertos, una vez hecho esto ya procedemos a hacer la conexión llamando al método conectar Puerto

**Método compararPuertos (retorna una cadena de caracteres)**

Este método retorna el nombre del nuevo puerto detectado, con unos bucles FOR esta comparado el contenido de los arreglos puertosNuevos y nombresPuertos hasta encontrar el nuevo Puerto, que en este caso sería el del dispositivo conectado, este método es llamado desde el método comprobarPuertos , para asignar a una variable el nombre del nuevo puerto.

**Método conectarPuerto**

Este método como lo indica su nombre , nos permite realizar la conexión entre nuestro programa y el puerto para poder realizar la comunicación, se crea una instancia de la clase SerialPort (que la provee la librería JSSC) enviándole como paramnetro la cadena con el nombre del puerto nuevo, si se puede abir el puerto entonces se inicializa la conexión pasándole los parámetros como el BaudRate, el DataBits etc.

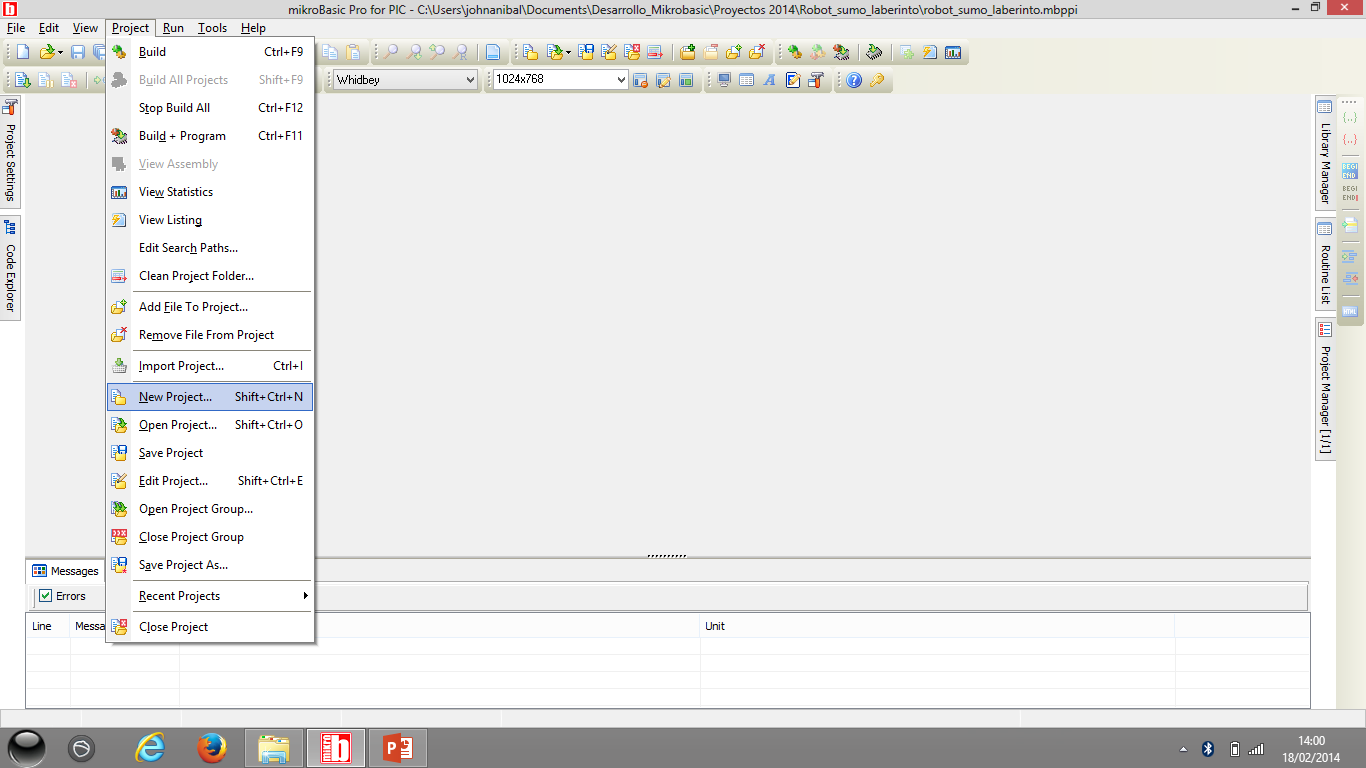
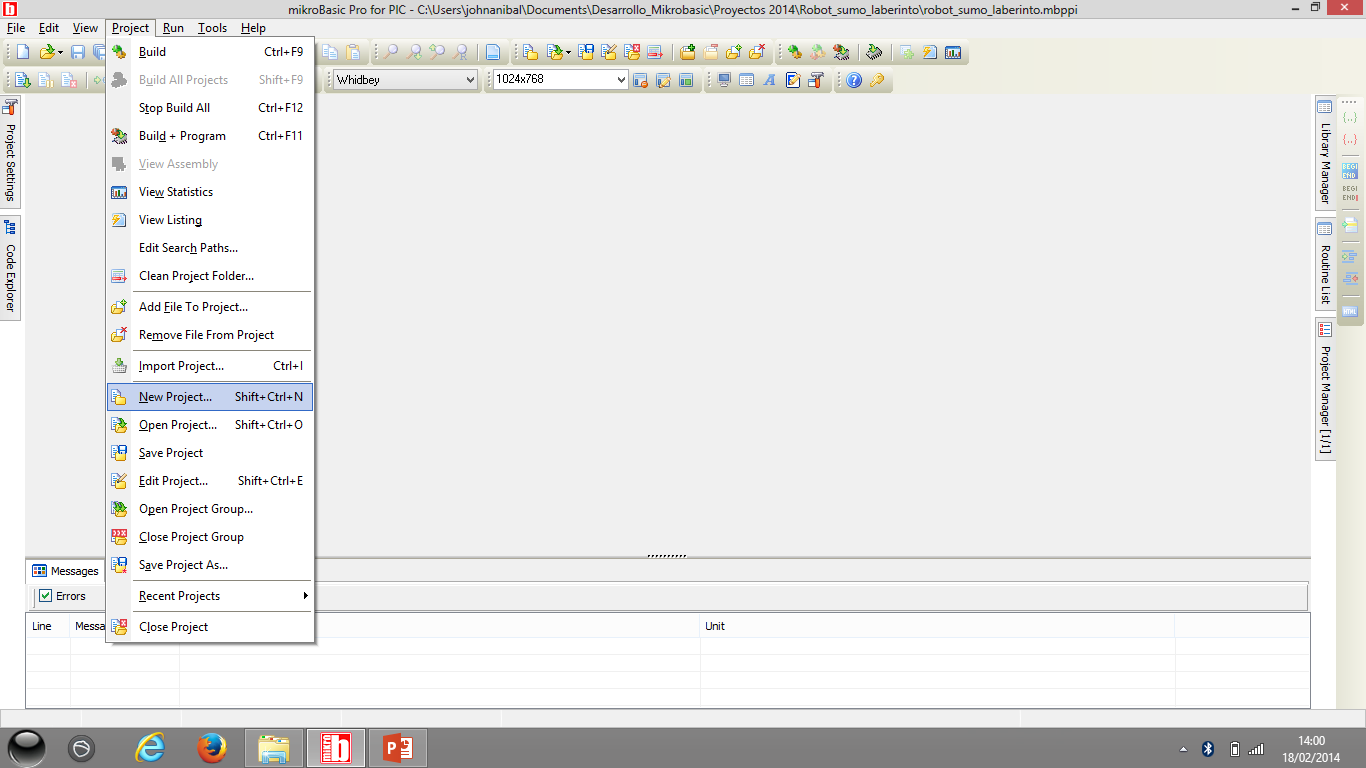
**ANEXOS**

1. **CREACION DE UN PROYECTO EN MIKROBASIC PRO FOR PIC**

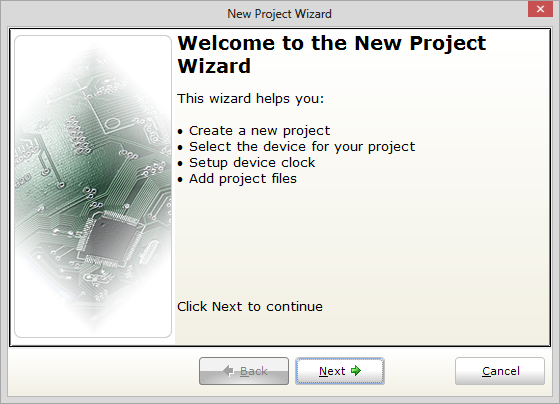
PASO # 1 EJECUTAR COMO ADMINISTRADOR MIKROBASIC PRO FOR PIC



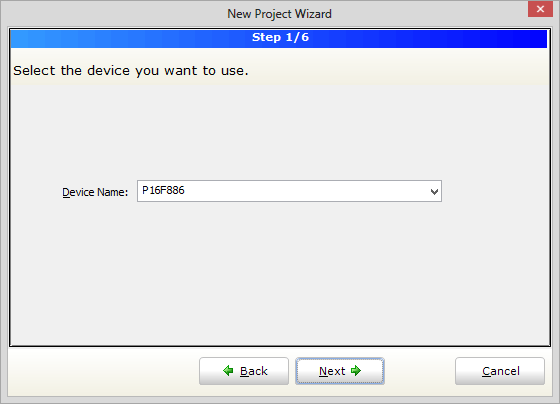
PASO # 2 PROJECT –NEW PROJECT



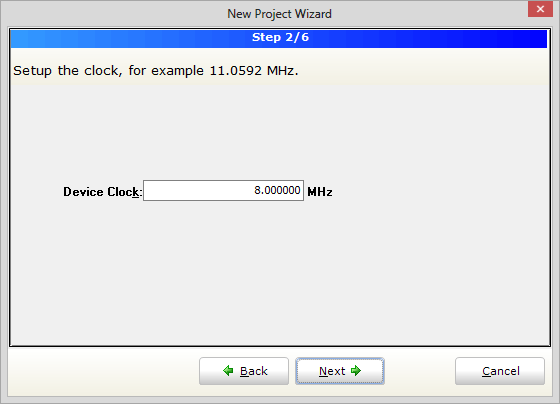
PASO # 3 CLICK EN NEXT



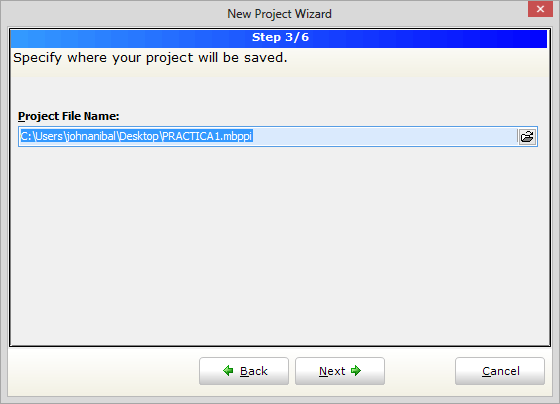
PASO # 4 DEVICE NAME: 16F886 -CLICK NEXT



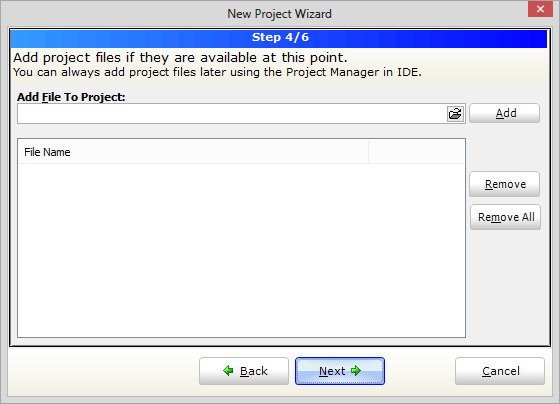
PASO # 5 DEVICE CLOCK: 8.000000 -CLICK NEXT



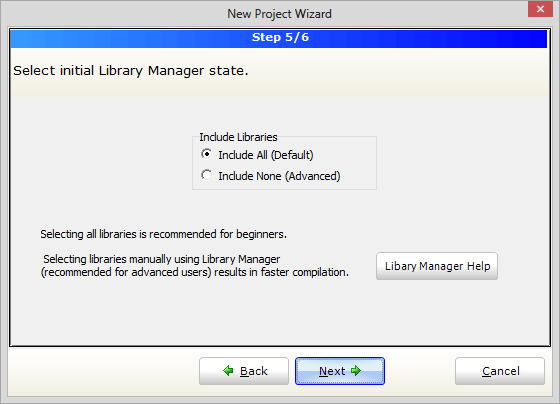
PASO # 6 GUARDAR EL ARCHIVO -CLICK NEXT



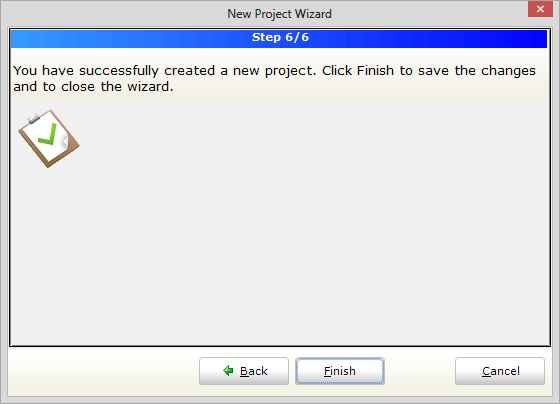
PASO # 7 CLICK NEXT



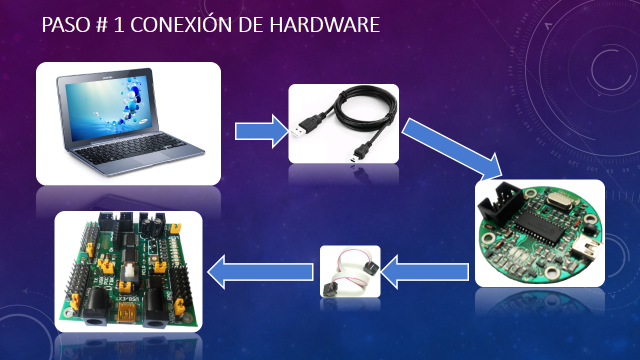
PASO # 8 CLICK NEXT



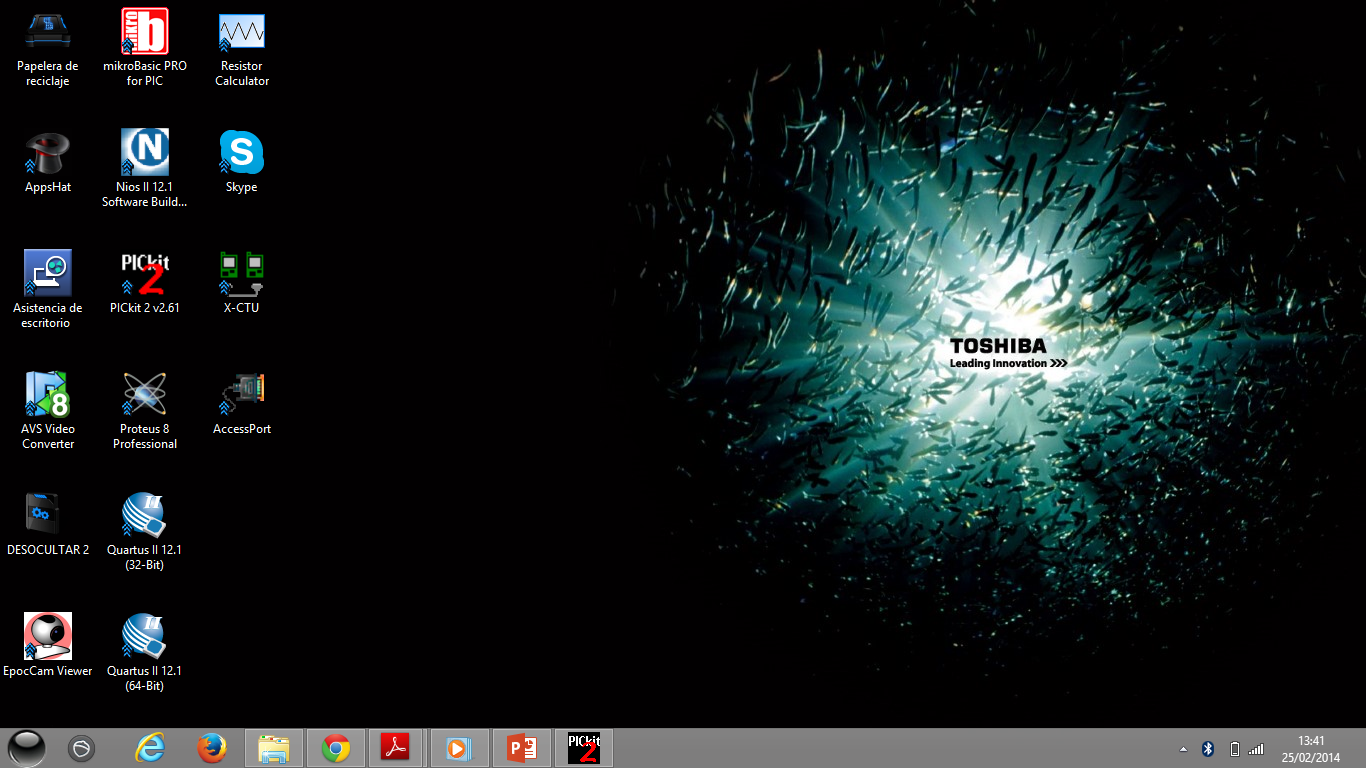
PASO # 9 CLICK NEXT



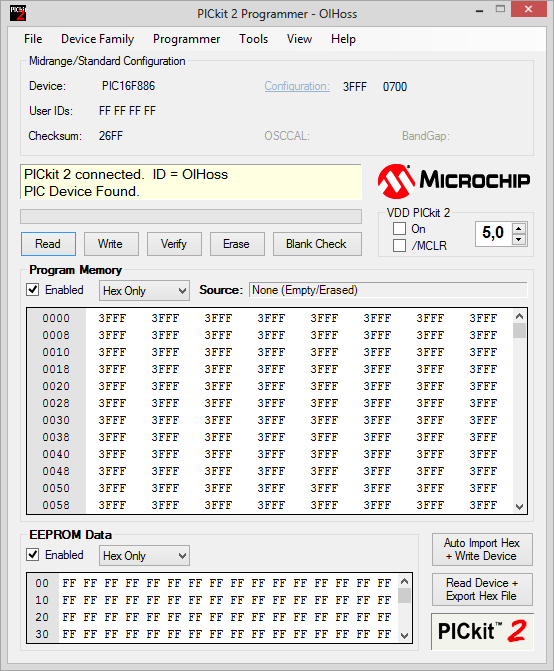
1. **PROGRAMACION DE UN MICROCONTROLADOR CON PICKIT2**

****

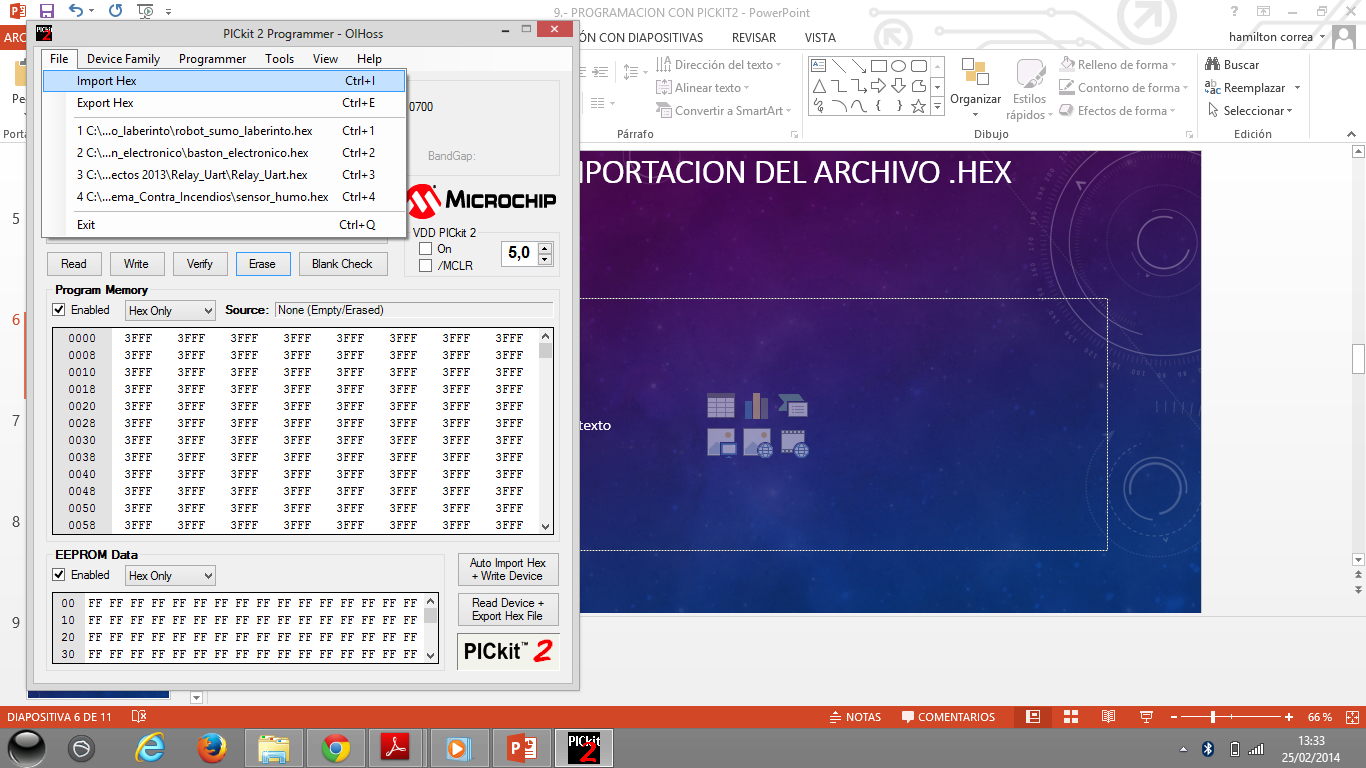
Ejecute el icono de PICKit

****

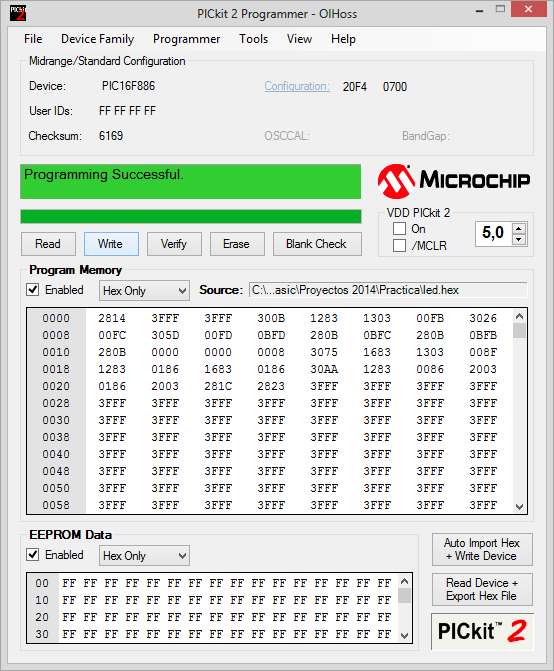
Hacer click en menu tool – check comunication

****

Observe que el dispositivo haya sido detectado y proceda a borrar la información con el botón de ERASE.

Importe el archivo .HEX 

Programe el dispositivo haciendo click en el botón de WRITE



**FUENTE BILBIOGRÁFICAS DE ANEXOS**

[**www.ideastechnology.com**](http://www.ideastechnology.com)