2012

# Voice control for a smart house

Høgskolen i Østfold

Oscar Andrés Bracho Pozo

04/06/2012

## Preface:

My name is Oscar Andres Bracho Pozo, student of electronic engineering at the University of Castilla La Mancha (Spain). This semester I have worked in a project relative with home automation specifically with the voice control at the University of Østfold University College.

I would like to thank my supervisors Helge E Mordt and Reidar Norby for the time, interest and the help received in the realization of this project.

# Index

# 1. Summary

In this section we will describe the main chapters in the project:

**1 .Introduction:** This section we put the purpose of the project and a brief preparation for the text that coming next.
**2. Problem definition:** There are the most important problems that we have in the project.
**3. Theory and tools:** theory and tools used in the creation of the project.
**4. Programming and practical part**: integration process in language of programming Visual Basic, of all the devices in the project
**5. Interface**: detailed explanation of the user interface
**6. Discussion:** The further work and some ideas for future students.
**7. Conclusion:** Final conclusion of the project
**8. Appendix:** Documents, plans and routine followed in this project.

# 2. Introduction.

The objective of the smart home is to provide a better quality of life by increasing the comfort, safety and comfort by increasing the number of services available useful for the inhabitant of the house.

Automatic speech recognition is a part of Artificial Intelligence that aims to enable voice communication between humans and electronic computers. A voice recognition system is a computational tool that can process the voice signal emitted by the humans and recognize the information contained in this.

This project was focussed on programming and development of new applications of speech recognition, for application in the field of home automation and smart homes.

## 2.1 Purpose.

The purpose of this project is design a program that can control multiples applications in a house, facilitating the mobility and improving the security and the comfort for people with physical disabilities.

The secondary purpose of this project is to demonstrate that is possible implement the basic home automation applications with a low cost, using the programs that we have in our computer, in the case of this project we used “Visual Basic 2010” and “Speech control” of windows 7.

The technical report is the communication unit itself .Contain all data about the construction process, problems solved, components, and future improvements. You can find everything related to the process carried out in the project called “Voice control for a smart house

# 3. Problem Definition.

The process of learning and adaptation of standard devices to be able to control lights and doors in a smart home

The main problem in the realization of this project was the distribution of time, followed the book "How design your own PC voice control" but the concepts were a bit old, and programs that were mentioned, their versions do not exist in the current market, so it has used the current versions for adapt to the project.

Visual basic

- Understand what the project is about.
- Learn the basic commands
- Design the program interface.
- Connection with peripheral devices.
- Connection with "speech control" of windows 7

Experimental interface board k8055 Velleman.

- Physical connection with the board
- Programming and creation of auxiliary modules for correct operation
- Study and adaptation of the output signals and input.

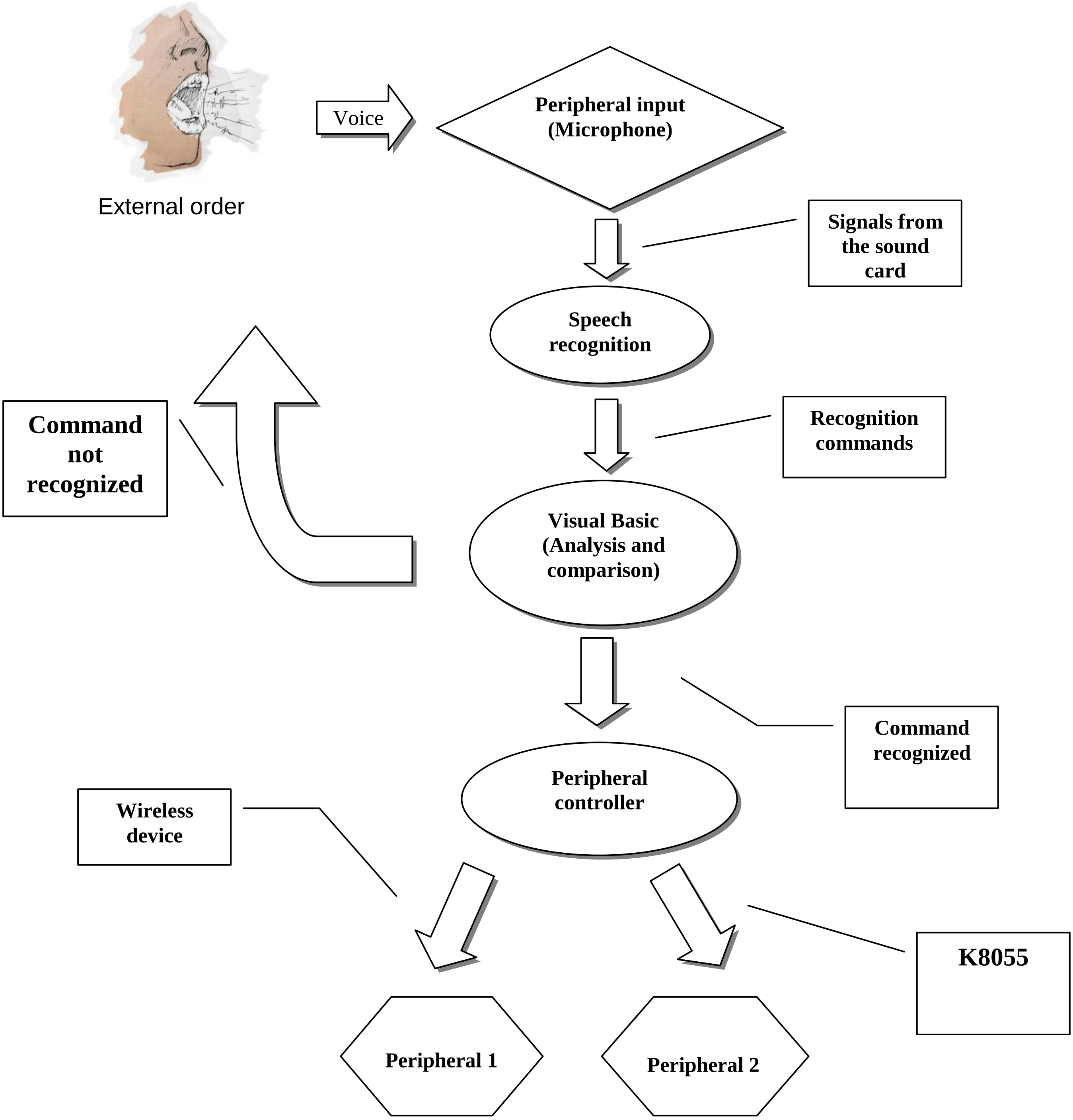
Tellstick

- Wireless configuration

- Configuration and coordination with Visual Basic and Speech Control
- Adaptation in the user interface.

## 4. Theory and tools.

In this section we will focus on the theory that has been used for the creation of this project. To make easier the understanding of this process the next diagram and then I will describe it.



*(Picture 1)*
*Operating diagram*

## 4.1 External order

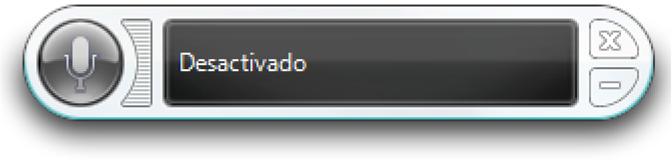
In this project orders or external commands are sent by the human voice, can also be controlled using the user interface on the computer, which was designed to recognize the situation of each peripheral component.

## 4.2 Microphone

The microphone is our peripheral input; the process of this device is to translate the vibrations due to acoustic pressure exerted on its capsule by the sound waves into electrical energy. In our case the microphone that we used it was the microphone standard in a normal computer.
The sound input process is administered by the audio card that allows us to recognize these electrical signals by the voice recognition system “Windows Speech Recognition” that is built into the operating system Windows 7.

## 4.3 Windows Speech Recognition

Is a speech recognition application included in Windows 7.This program is responsible to recognize the order and transmit it to the Visual Basic engine, is one of the most important intermediates in the communication between human/computer.


Desactivado

*(Picture 2)*
*Speech recognition interface of windows 7*

The voice recognition system of Windows is a quite useful tool to navigate in the OS Windows 7, below show a few examples which have been used in the project.

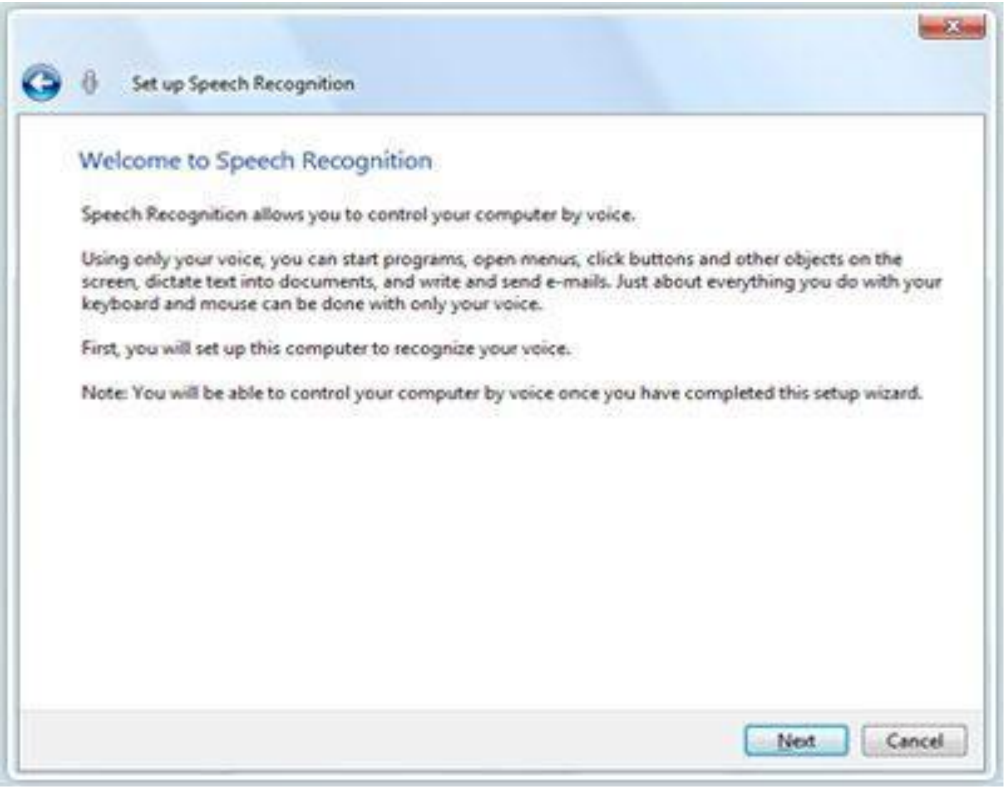
The next guide was used to configure the speech recognition system in windows 7.

*Before you can use the speech feature, there are some things that must be done. Unlike keyboards and mice, where the finger pressure is converted into a constant electronic pressure, voice has variations. So the software must be trained.*

•*First of all, there is a hardware issue. One needs a very good microphone. Avoid cheaper microphones since they tend to pick up a lot of background noise.*

•*Secondly create a voice profile and train the computer to identify the speaker's voice and pronunciation. There is a speech training tutorial*.

- Click the Start button
- Click Control Panel
- Click Ease of Access
- Click Speech Recognition
- Click Take Speech Tutorial
- And finally, speak clearly and pronounce your words carefully, not too fast or too slow.



Set up Speech Recognition

Welcome to Speech Recognition

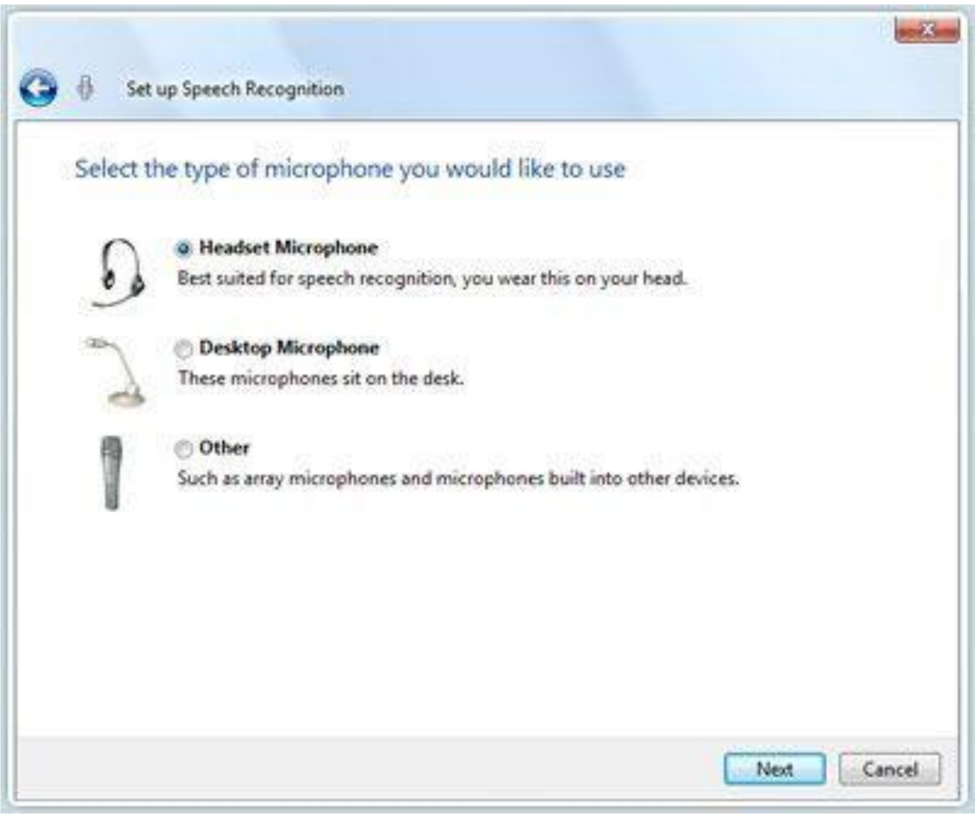
Speech Recognition allows you to control your computer by voice.

Using only your voice, you can start programs, open menus, click buttons and other objects on the screen, dictate text into documents, and write and send e-mails. Just about everything you do with your keyboard and mouse can be done with only your voice.

First, you will set up this computer to recognize your voice.

Note: You will be able to control your computer by voice once you have completed this setup wizard.

*(Picture 3)*
*Picture extracted from Guide "How to Use Speech Recognition with Windows 7"*



Set up Speech Recognition

Select the type of microphone you would like to use

● **Headset Microphone**
Best suited for speech recognition, you wear this on your head.

○ **Desktop Microphone**
These microphones sit on the desk.

○ **Other**
Such as array microphones and microphones built into other devices.

*(Picture 4)*
*Picture extracted from Guide "How to Use Speech Recognition with Windows 7"*

*Key Features of Speech Recognition*
*Speech recognition is separated into the following categories:*

- ***Multiple languages:*** *Windows Speech Recognition in Windows 7 is available in eight languages/dialects: English, French, Spanish, German, Japanese, Simplified Chinese, and Traditional Chinese.*

- ***"How do I" help****: If one says "How do I…" do, or make, or find … a task that one wants to perform with the computer, the software will respond accordingly. For example, "How do I change my desktop theme?"*

- ***Correction:*** *One is able to fix incorrectly recognized words by selecting from alternatives provided for the dictated phrase or word, or by spelling the word.*

- ***Dictation:*** *Address e-mails and documents by dictation; make corrections, and save the work.*

- ***Adaptation****: Modification and adaptation to speaking styles and accents. The software learns and improves the interaction with the computer*

- ***Command****: This is the "Say what you see" feature. It enables one to naturally control applications and complete tasks. Among such them are launching documents or applications, formatting, saving or deleting documents.*

- ***Disambiguation:*** *Ambiguous words and phrasings can be resolved with a user interface for clarification. If something can be misinterpreted, the software can clarify it.*

*Text from guide "How to Use Speech Recognition with Windows 7"*

When the configuration process is finished we can focus on adapting this application in the main program.

## 4.4 Visual Basic.

Visual basic was the main program used in the creation of this project, this program has the ability to adapt to other programs, making it much easier to program and integrate the various devices peripherals.

We can describe Visual Basic as the engine of the project, because in it the orders and commands are compared, processed and resolved. In case of acceptance the order, visual basic is responsible for answering the user, indicating that the order was carried out, in the opposite case the program will prompt the user that repeat the order.

## 4.5 Peripheral controllers

Peripheral controllers are responsible for sending the right signals to perform a certain action, in this project have used the wireless system called "*Tellstick*" and the experimental board "*The K8055 interface board*" which are described below.

### 4.5.1 Wireless system "*Tellstick*"

This wireless control device was developed specifically for the field of home automation, was designed for the control by radio of different devices, using any device that can support the USB system.
TellStick transmits signals at 433.92 MHz that turns on/off electricity on electronics & lights connected to wall-plug socket receivers.

*(Picture 5)*
*Wireless device TellStick*

### 4.5.2 Experimental board the K8055 interface board

*(Picture 6)*
*Experimental Board K8055 Velleman*

Next we will define the device features, most common commands and its work on the project.

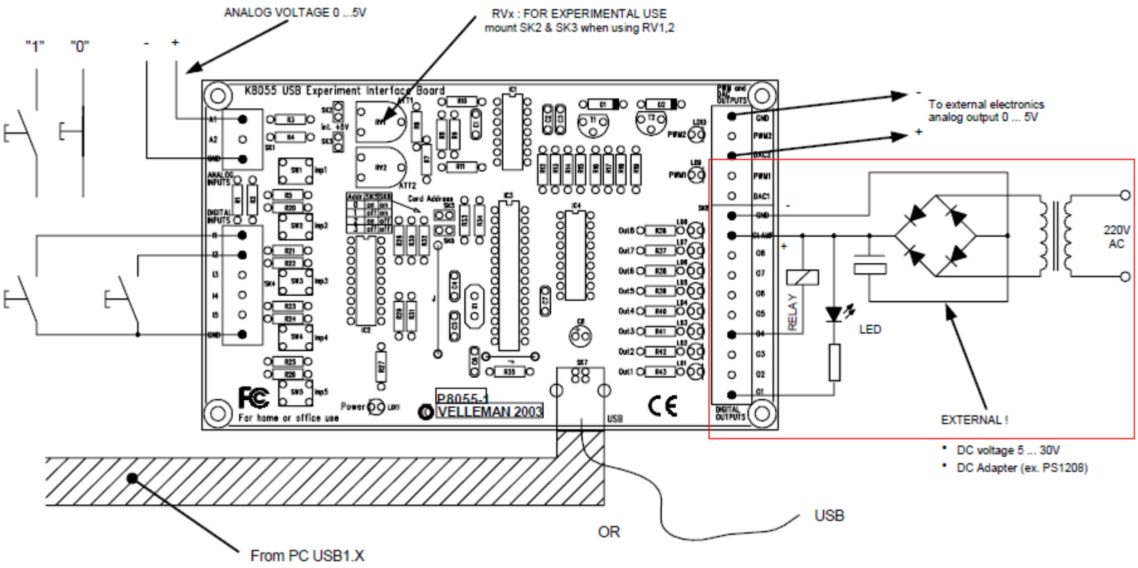
*The K8055 interface board has 5 digital input channels and 8 digital output channels. In addition, there are two analogue inputs and two analogue outputs with 8 bit resolution. The number of inputs/outputs can be further expanded by connecting more (up to a maximum of four) cards to the PC's USB connectors. All communication routines are contained in a Dynamic Link Library (DLL). You may write custom Windows (98SE, 2000, Me, XP) applications in Delphi, Visual Basic, C++ Builder or any other 32-bit Windows application development tool that supports calls to a DLL.*

*Features*

*DIAGNOSTIC / TEST SOFTWARE*
*:*

- *separate output / input test*
- *clear all / set all function*
- *counter function on inputs 1 and 2 with adjustable debounce (max 2kHz depends on total I/O load)*
- *analogue output set sliders*
- *analogue input bar-graph indication*

This board will be responsible for sending the orders to the peripheral devices, in this case is a door connected to the electrical system, consequently the output of the board must have the following configuration.



*(Picture 7)*
*Picture extract from datasheet of Experimental Board K8055 Velleman,*

The configuration at the output of the device will help the proper functioning of the final process and also will affect in the protection of the board.

## 4.6 Peripherals

The peripheral devices that we use in this project are:

*(Picture 8)*
*Floor lamp and control device Nexa 2300w*

A lamp controlled by the device Nexa 2300W

*(Picture 8)*
*Common door*

And in the expo will be used a gate connected to an engine controlled by the experimental board K8055.

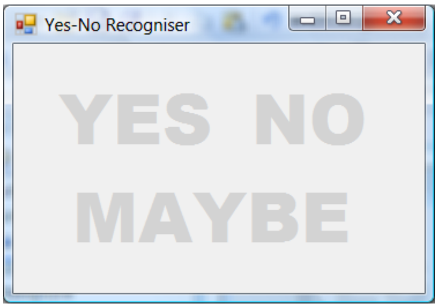
# 5. Programming and practical part.

This section explains everything related to programming of the components that make up the project to give a global vision of the integration process, in relation to the programming, will focus directly on explaining the steps taken to carry out the project.
This integration process we will focus the program Visual Basic, first we introduce to the programming environment of Visual Basic with a simple example which will be explained as follows.

## 5.1 Adaptation to the programming environment Visual Basic.NET

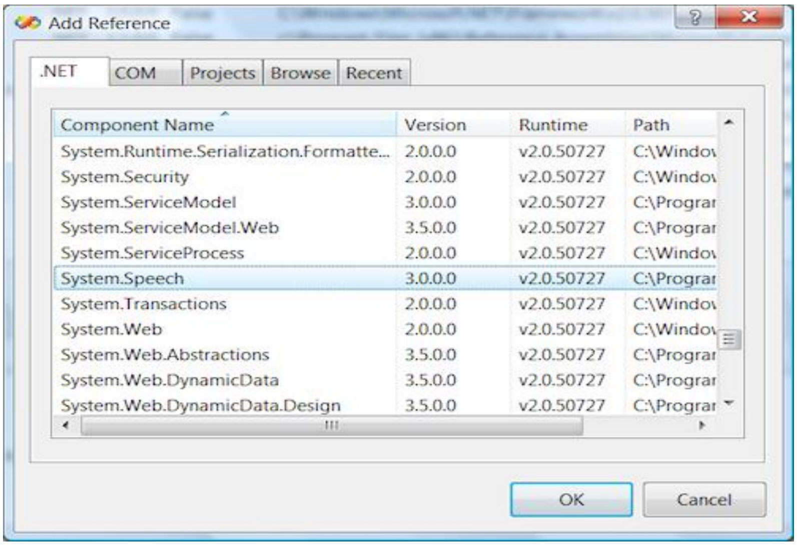
How to adapt the voice control system in Visual Basic .NET

***A "Yes-No-Maybe" application***
*Create a Windows Forms application. Give the application three labels: LabelYes, LabelNo and LabelMaybe. Initialise these to a large font and a light-gray colour:*



*(Picture 9)*
*Interface of the example speech recognition*

***Reference the System.Speech component***
*Use Project|Add Reference to add a reference to the System.Speech component:*

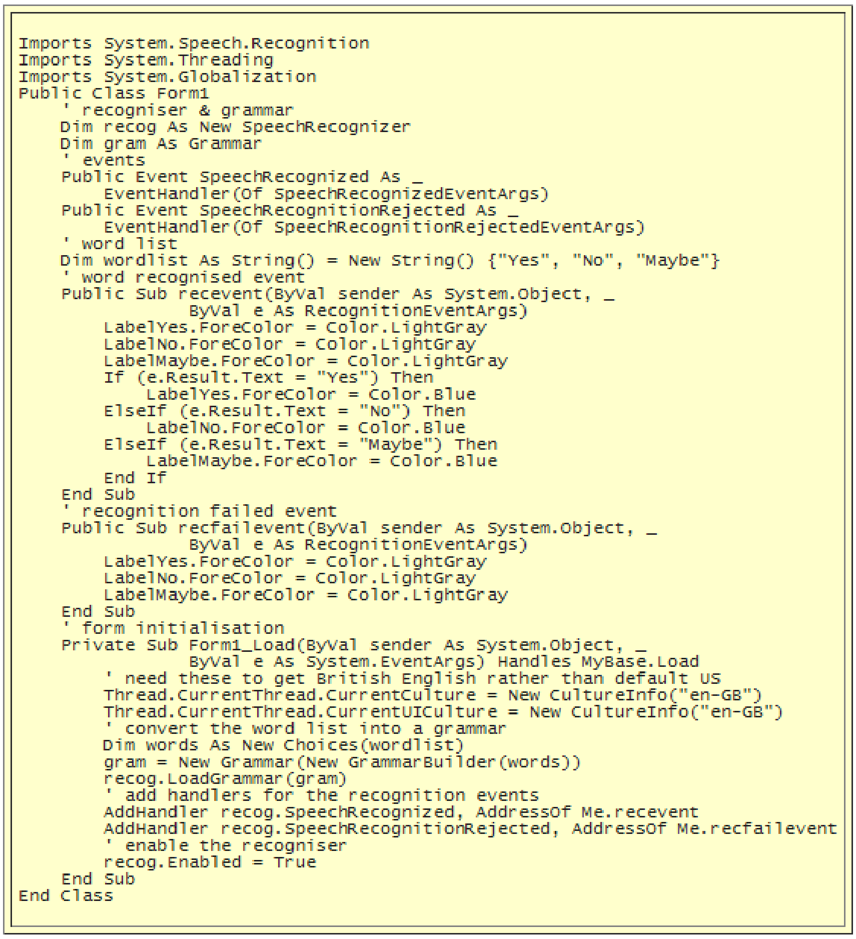


| Component Name | Version | Runtime | Path |
|---|---|---|---|
| System.Runtime.Serialization.Formatte... | 2.0.0.0 | v2.0.50727 | C:\Windov |
| System.Security | 2.0.0.0 | v2.0.50727 | C:\Windov |
| System.ServiceModel | 3.0.0.0 | v2.0.50727 | C:\Prograr |
| System.ServiceModel.Web | 3.5.0.0 | v2.0.50727 | C:\Prograr |
| System.ServiceProcess | 2.0.0.0 | v2.0.50727 | C:\Windov |
| System.Speech | 3.0.0.0 | v2.0.50727 | C:\Prograr |
| System.Transactions | 2.0.0.0 | v2.0.50727 | C:\Windov |
| System.Web | 2.0.0.0 | v2.0.50727 | C:\Windov |
| System.Web.Abstractions | 3.5.0.0 | v2.0.50727 | C:\Prograr |
| System.Web.DynamicData | 3.5.0.0 | v2.0.50727 | C:\Prograr |
| System.Web.DynamicData.Design | 3.5.0.0 | v2.0.50727 | C:\Prograr |

*(Picture 10)*
*Process of adaptation in visual basic, of the voice recognition software*

## ***Recognition code***

*Add this code to the form:*



```
Imports System.Speech.Recognition
Imports System.Threading
Imports System.Globalization
Public Class Form1
    ' recogniser & grammar
    Dim recog As New SpeechRecognizer
    Dim gram As Grammar
    ' events
    Public Event SpeechRecognized As _
        EventHandler(Of SpeechRecognizedEventArgs)
    Public Event SpeechRecognitionRejected As _
        EventHandler(Of SpeechRecognitionRejectedEventArgs)
    ' word list
    Dim wordlist As String() = New String() {"Yes", "No", "Maybe"}
    ' word recognised event
    Public Sub recevent(ByVal sender As System.Object, _
            ByVal e As RecognitionEventArgs)
        LabelYes.ForeColor = Color.LightGray
        LabelNo.ForeColor = Color.LightGray
        LabelMaybe.ForeColor = Color.LightGray
        If (e.Result.Text = "Yes") Then
            LabelYes.ForeColor = Color.Blue
        ElseIf (e.Result.Text = "No") Then
            LabelNo.ForeColor = Color.Blue
        ElseIf (e.Result.Text = "Maybe") Then
            LabelMaybe.ForeColor = Color.Blue
        End If
    End Sub
    ' recognition failed event
    Public Sub recfailevent(ByVal sender As System.Object, _
            ByVal e As RecognitionEventArgs)
        LabelYes.ForeColor = Color.LightGray
        LabelNo.ForeColor = Color.LightGray
        LabelMaybe.ForeColor = Color.LightGray
    End Sub
    ' form initialisation
    Private Sub Form1_Load(ByVal sender As System.Object, _
            ByVal e As System.EventArgs) Handles MyBase.Load
        ' need these to get British English rather than default US
        Thread.CurrentThread.CurrentCulture = New CultureInfo("en-GB")
        Thread.CurrentThread.CurrentUICulture = New CultureInfo("en-GB")
        ' convert the word list into a grammar
        Dim words As New Choices(wordlist)
        gram = New Grammar(New GrammarBuilder(words))
        recog.LoadGrammar(gram)
        ' add handlers for the recognition events
        AddHandler recog.SpeechRecognized, AddressOf Me.recevent
        AddHandler recog.SpeechRecognitionRejected, AddressOf Me.recfailevent
        ' enable the recogniser
        recog.Enabled = True
    End Sub
End Class
```

*(Picture 11)*
*Example of speech recognition in visual basic*

***Run the application***
*When the application starts the Windows speech recognition system will be loaded. You will need to say "Start Listening" or to click on the microphone icon to start recognition.*

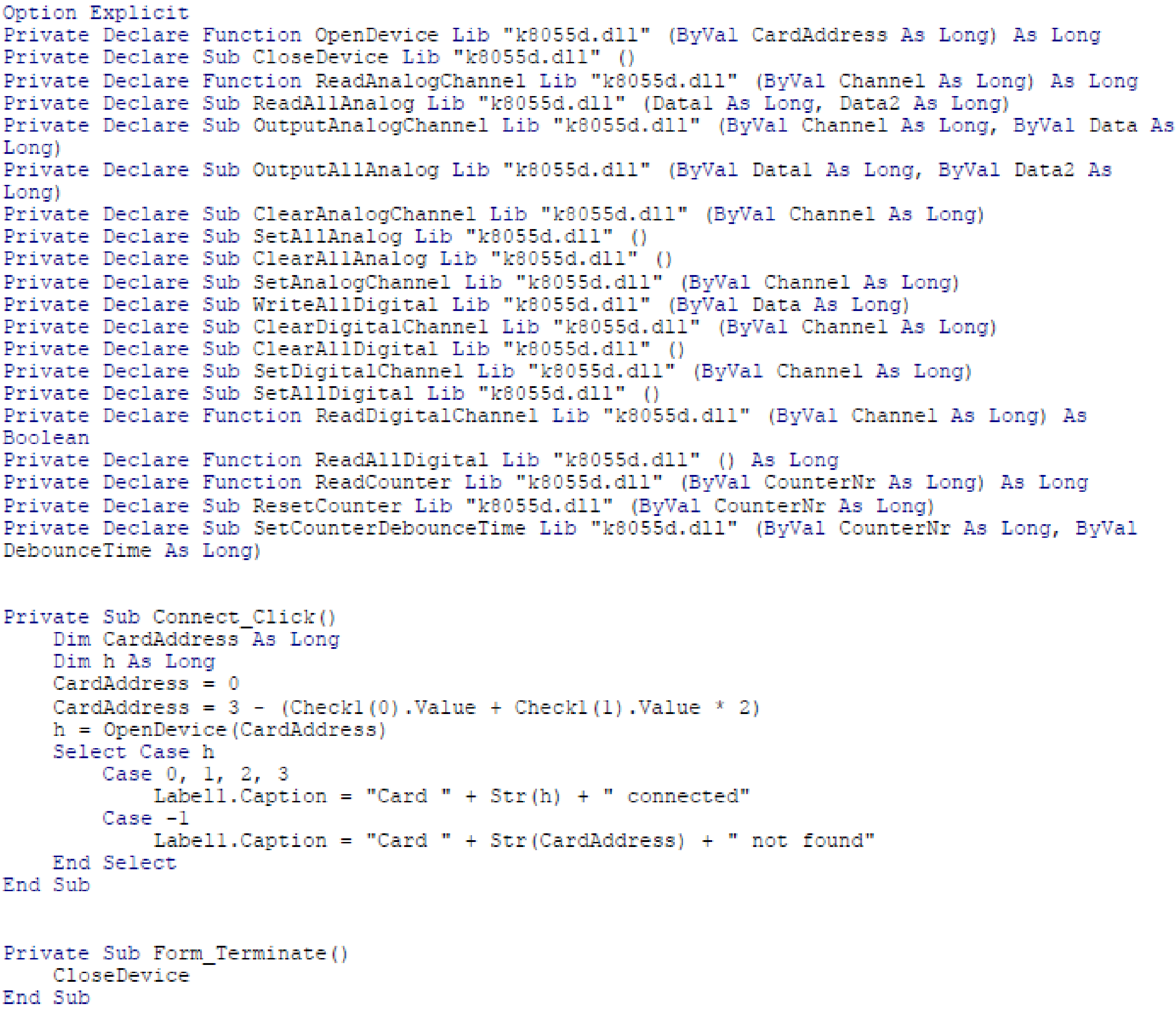
*Then when you say "yes", "no" or "maybe", the appropriate label will light up. If you say anything else, the labels should turn back to grey.*
*To improve recognition, you should set up the speech recognition system for your microphone, environment and your voice using the Speech Recognition applet in the Control Panel.*
*Text from guide "Computational Methods for Research in Speech Science"*

This example is an advance in integrating the visual basic application with the speech recognition, following the above steps we can practice and learn a little more code with which we will work.

Another important integration process is the peripheral controllers

For connecting the experimental board K8055 Velleman, and correct operation with the main program, must take the following steps.
*In the listing of an application example there are declarations of K8055D.DLL procedures and functions and an example hot to use the two more important DLL function calls: OpenDevice and CloseDevice*

***Note:** Make sure that the fileK8055D.DLL is copied to Windows' SYSTEM32 folder:*
Following the instructions in the datasheet of the experimental board, created in the main program of Visual Basic a module that contains these lines of code, with which we can control this device.



```
Option Explicit
Private Declare Function OpenDevice Lib "k8055d.dll" (ByVal CardAddress As Long) As Long
Private Declare Sub CloseDevice Lib "k8055d.dll" ()
Private Declare Function ReadAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long) As Long
Private Declare Sub ReadAllAnalog Lib "k8055d.dll" (Data1 As Long, Data2 As Long)
Private Declare Sub OutputAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long, ByVal Data As
Long)
Private Declare Sub OutputAllAnalog Lib "k8055d.dll" (ByVal Data1 As Long, ByVal Data2 As
Long)
Private Declare Sub ClearAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub SetAllAnalog Lib "k8055d.dll" ()
Private Declare Sub ClearAllAnalog Lib "k8055d.dll" ()
Private Declare Sub SetAnalogChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub WriteAllDigital Lib "k8055d.dll" (ByVal Data As Long)
Private Declare Sub ClearDigitalChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub ClearAllDigital Lib "k8055d.dll" ()
Private Declare Sub SetDigitalChannel Lib "k8055d.dll" (ByVal Channel As Long)
Private Declare Sub SetAllDigital Lib "k8055d.dll" ()
Private Declare Function ReadDigitalChannel Lib "k8055d.dll" (ByVal Channel As Long) As
Boolean
Private Declare Function ReadAllDigital Lib "k8055d.dll" () As Long
Private Declare Function ReadCounter Lib "k8055d.dll" (ByVal CounterNr As Long) As Long
Private Declare Sub ResetCounter Lib "k8055d.dll" (ByVal CounterNr As Long)
Private Declare Sub SetCounterDebounceTime Lib "k8055d.dll" (ByVal CounterNr As Long, ByVal
DebounceTime As Long)


Private Sub Connect_Click()
    Dim CardAddress As Long
    Dim h As Long
    CardAddress = 0
    CardAddress = 3 - (Check1(0).Value + Check1(1).Value * 2)
    h = OpenDevice(CardAddress)
    Select Case h
        Case 0, 1, 2, 3
            Label1.Caption = "Card " + Str(h) + " connected"
        Case -1
            Label1.Caption = "Card " + Str(CardAddress) + " not found"
    End Select
End Sub


Private Sub Form_Terminate()
    CloseDevice
End Sub
```

*(Picture 12)*
*Code to visual basic adaptation of K8055*

We will make the same procedure to adapt the USB wireless controller. In this case use a different module, but with the same utility than the previous

```
Module Module2
    Public Declare Function tdTurnOn Lib "TelldusCore.dll" (ByVal lngDeviceId As Integer) As Boolean
    Public Declare Function tdTurnOff Lib "TelldusCore.dll" (ByVal lngDeviceId As Integer) As Boolean
    Public Declare Function tdBell Lib "TelldusCore.dll" (ByVal lngDeviceId As Integer) As Boolean
    Public Declare Function tdDim Lib "TelldusCore.dll" (ByVal lngDeviceId As Integer, ByVal level As
Byte) As Boolean
    Public Declare Function tdMethods Lib "TelldusCore.dll" (ByVal lngDeviceId As Integer, ByVal
methodsSupported As Short) As Integer
    Public Declare Function tdGetDeviceId Lib "TelldusCore.dll" (ByVal a As Integer) As Integer
    Public Declare Function tdGetName Lib "TelldusCore.dll" (ByVal i As Integer) As String
    Public Declare Function tdGetNumberOfDevices Lib "TelldusCore.dll" () As Integer
End Module
```

The most important commands in this module are those that allow us to turn on and off the lights, to be more specific:

```
Public Declare Function tdTurnOn Lib "TelldusCore.dll" (ByVal lngDeviceId As Integer) As Boolean
Public Declare Function tdTurnOff Lib "TelldusCore.dll" (ByVal lngDeviceId As Integer) As Boolean
```

Must be emphasized that the functions were declared as public functions, because you can invoke these functions from anywhere in the project.

Main program.
The next step will be to develop the main program:

```
Option Explicit On
Imports System.Speech.Recognition
Imports System.Speech.Recognition.SrgsGrammar
```

With the code we can see that we import the voice recognition system

Applications use *System.Speech.Recognition* name space to access extend this basic speech recognition technology, by defining algorithms for identifying and acting on specific phrases or word patterns, and by managing the run time behavior of this speech infrastructure.

Grammars created with members of the *System.Speech.Recognition.SrgsGrammar* namespace can be used by constructors of the Grammar class to create Grammar objects.

The following lines of code, we show in detail the usefulness of each function, as you can see the lines of green text gives us a clear idea of the usefulness of the functions.

```
Public Class Form1k8055

    ''PUBLIC VARIABLE OBJECT SPEECHRECOGNIZER
    Public recognizer As SpeechRecognizer
    ''VARIABLE TO REPRODUCE TEXT
    Public voz As New Speech.Synthesis.SpeechSynthesizer

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        ''INITIALIZE THE VARIABLE THAT WE USE FOR RECOGNITION
        recognizer = New SpeechRecognizer()
        ''WE ADD RECOGNITION EVENT
        AddHandler recognizer.SpeechDetected, AddressOf detectado
        AddHandler recognizer.SpeechRecognitionRejected, AddressOf noreco
        AddHandler recognizer.SpeechRecognized, AddressOf reco
        ''We indicate that the object is A = TRUE, ACTIVATED
        recognizer.Enabled = True
        ''DECLARE OBJECT GRAMMAR
        Dim grammar As New GrammarBuilder
        ''WE ADD PHRASES TO RECOGNIZE
        grammar.Append(New Choices("light_on", "light_off", "open_door", "close_door", _
        "reset", "More_light", "Less_light", "Computer", "Clean", "Delete"))
        ''CHARGE THE OBJECT GRAMMAR IN THE OBJECT SPEECHRECOGNIZER
        recognizer.LoadGrammar(New Grammar(grammar))
        System.Windows.Forms.Application.DoEvents()
    End Sub
    ''EVENTS OF SYSTEM.SPEECH.RECOGNITION
    ''SPEECH IS DETECTED BY MICROPHONE
    Private Sub detectado(ByVal sender As Object, ByVal e As SpeechDetectedEventArgs)

        ''MsgBox(e.AudioPosition.Duration.ToString())
    End Sub
    Private Sub noreco(ByVal sender As Object, ByVal e As
SpeechRecognitionRejectedEventArgs)
        ''MsgBox("not recognized the comand de Voice")
        voz.Speak("The programm does not find the command, repeat the order please")
    End Sub
```

Next we find the second part of the main program; the code shows the cases that can be presented and actions for each of them

```
Private Sub reco(ByVal sender As Object, ByVal e As SpeechRecognizedEventArgs)

        Select Case e.Result.Text.ToUpper.ToString
            ''POSSIBLE CASES
            Case "LIGHT_ON"
                tdTurnOn(1)
                voz.Speak("The lights are lit")
                SetDigitalChannel(1)
                ClearDigitalChannel(2)
                TextBox1.Text &= 1

            Case "LIGHT_OFF"
                tdTurnOff(1)
                voz.Speak("The lights are off")
                SetDigitalChannel(2)
                ClearDigitalChannel(1)
                TextBox1.Text &= 2

            Case "OPEN_DOOR"
                voz.Speak("The door is opening")
                SetDigitalChannel(3)
                ClearDigitalChannel(1)
                ClearDigitalChannel(2)
                ClearDigitalChannel(4)
                TextBox1.Text &= 3

            Case "CLOSE_DOOR"
                voz.Speak("The door is closing")
                SetDigitalChannel(4)
                ClearDigitalChannel(3)
                TextBox1.Text &= 4
            Case "RESET"
                voz.Speak("The  system is rebooting")
                ClearDigitalChannel(1)
                ClearDigitalChannel(2)
                ClearDigitalChannel(3)
                ClearDigitalChannel(4)
                ClearDigitalChannel(5)
                ClearDigitalChannel(6)
                ClearDigitalChannel(7)
                ClearDigitalChannel(8)
                ClearDigitalChannel(9)
                TextBox1.Text &= 5
                voz.Speak("the system is restarted")
            Case "MORE_LIGHT"
                SetDigitalChannel(5)
            Case "LESS_LIGHT"
                SetDigitalChannel(6)

            Case "CLEAN"
                TextBox1.Text = ""
                TextBox2.Text = ""
            Case Else

        End Select

    End Sub
```

The last part of the code shows us the situation of the experimented board K8055 Velleman, in the interface of Visual basic, we can see if the board is connected or not in the interface that will be explain in the next point. The code of the status is the next.

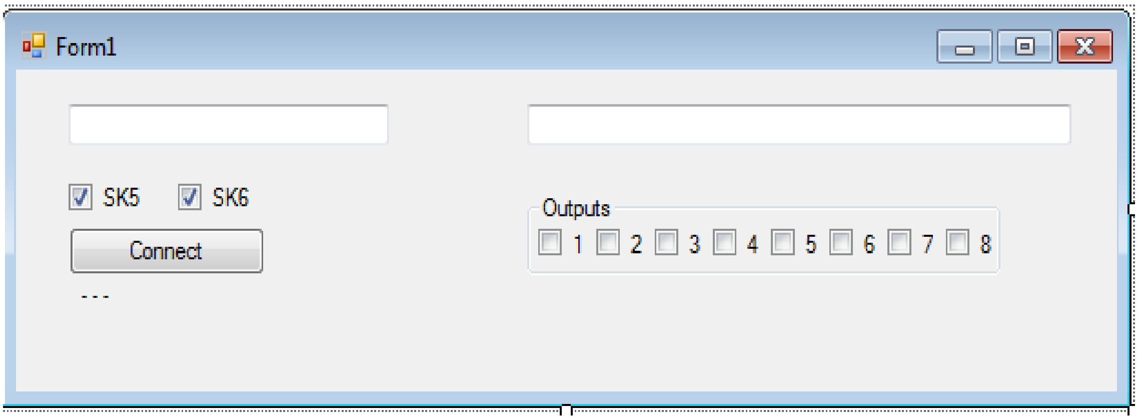
```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim CardAddress As Integer
        Dim h As Integer
        CardAddress = 3
        If CheckBox1.Checked Then CardAddress = CardAddress - 1
        If CheckBox2.Checked Then CardAddress = CardAddress - 2
        h = OpenDevice(CardAddress)
        Select Case h
            Case 0, 1, 2, 3
                Label1.Text = "Card " + Str(h) + " connected"
            Case -1
                Label1.Text = "Card " + Str(CardAddress) + " not found"
        End Select
        If h >= 0 Then Timer1.Enabled = True
    End Sub


    Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Label1.Click

    End Sub
End Class
```

# 6. Interface.

The interface that controlled the main program and shows the situation and problems with the peripherals is the next.



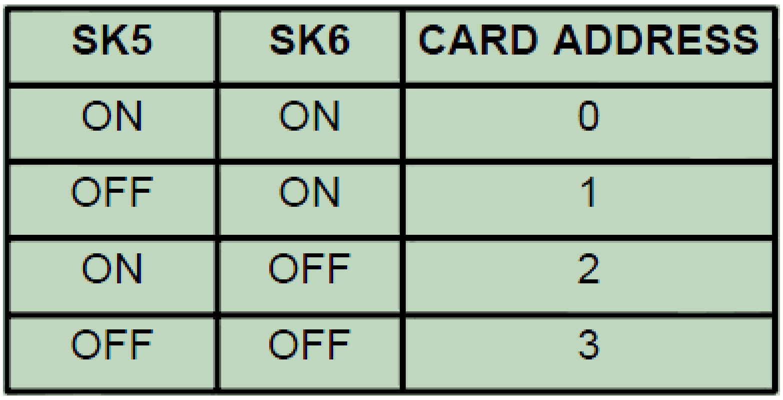
*(Picture 13)*
*Visual Basic interface of the main program*

The interface shows us the number of the order that we have said the number of the orders are as follows:

```
"LIGHT_ON"=  1;
"LIGHT_OFF"= 2;
"OPEN_DOOR"= 3;
"CLOSE_DOOR"=4;
"RESET"=     5;
"MORE_LIGHT"=6;
"LESS_LIGHT"=7;
```

The numbers appear on the box which is located in the upper left corner, depending on the order that we have given.

The boxes marked SK5 SK6 give us the address of the card, according to the following table.



| SK5 | SK6 | CARD ADDRESS |
|---|---|---|
| ON | ON | 0 |
| OFF | ON | 1 |
| ON | OFF | 2 |
| OFF | OFF | 3 |

**TABLE 1: Jumper SK5, SK6 Settings**

The button “Connect” its function is to connect the pc board, the status text that is below the button will indicate if the board is connected or not, is a quite useful function by allowing us to prevent cable failures or unexpected disconnections

The eight boxes unmarked indicate which outputs are active and which are not, this is useful to know the status of the outputs of our training board.

# 7. Discussion

I believe that this project is a good basis for future students who want to improve the quality and optimize the time for creating a project with these characteristics, smart house have enormous potential, and there are numerous opportunities to continue working on it.

## 7.1 Possible improvements

1. The interface design, make it more interactive for users
2. Add to the voice recognition system, Bluetooth as hands free, this wireless system is compatible with this project, and is able to improve its use for people with physical disabilities
3. Creation of specific control boards for this project, capable of replacing the experimental board K8055 and the wireless control system Tellstick.
4. Add more control systems, for example control of curtains, heating and creating a "safe mode" that will be responsible of locked the doors
5. Option to program by voice a mode "routine" and improve the comfort of home, for example turning on the heating 15 min before returning from the work.

# 8. Conclusion

The project has been complicated but rewarding, I learned a lot about of field of automation and I improving my domain in programs that I had not used before.

It is the beginning of a project with huge potential, if future students want continue with it.

The test done in the program, as expected has been satisfactory. I am very glad with the result.

# 9. References.

**Books**

Richard and Darren Harwood (2006*), Design your own PC Voice Control System* pg 41-160 Elektor.

**Webs**

Channel9 (2012*) Giving Computers a Voice* Retrieved 2012, June 4 from http://channel9.msdn.com/coding4fun/articles/Giving-Computers-a-Voice#c633238704000000000?areaType=Blogs&areaName=Coding4FunArticles

Espacio latino (2010) *Crea tu propia web* Retrieved 2012, June 4 from http://creatuweb.espaciolatino.com/tutorhtml/tema3.html

Msdn (2008) *Text-to-Speech Tutorial (SAPI 5.3)* Retrieved 2012, June 4 from http://msdn.microsoft.com/en-us/library/ms720163%28VS.85%29.aspx

SPEECH, HEARING & PHONETIC SCIENCES (2007) *How to Use Speech Recognition Within a Visual Basic. NET application* Retrieved 2012, June 4 from *http://www.phon.ucl.ac.uk/courses/spsci/compmeth/speech/recognition.html*

Electro Tech (2011) *K8055 USB Interface Visual Basic Help* Retrieved 2012, June 4 from http://www.electro-tech-online.com/general-electronics-chat/23652-k8055-usb-interface-visual-basic-help.html

Msdn (2008) *Introducción al lenguaje de programación Visual Basic* Retrieved 2012, June 4 from http://msdn.microsoft.com/es-es/library/xk24xdbe(v=vs.80).aspx

SPEECH, HEARING & PHONETIC SCIENCES(2007) *How to use Text-to-Speech within a Visual Basic .NET* application Retrieved 2012, June 4 from http://www.phon.ucl.ac.uk/courses/spsci/compmeth/speech/synthesis.html

Nexa Electronics *(2012) EYCR-2300 / Extra* Retrieved 2012, June 4 from http://www.nexa.se/EYCR-2300-Extra.htm

# APPENDIX

| Week | Activity | Goal | Res |
|---|---|---|---|
| 13 | Learn about Visual Basic | Understanding the programming environment | DONE |
| 14 | Work in the web page and pre report | Design a website and deliver the pre-report | DONE |
| 15 | Learn the programming | Introduction to speech recognition software from Microsoft | DONE |
| 16 | Develop the user interface | Visual Basic | DONE |
| 17 | Programming | Visual Basic | DONE |
| 18 | Programming | Visual Basic | DONE |
| 19 | Programming | Visual Basic | DELAYED (WEEK 20) |
| 20 | Testing | Voice control is working properly | DONE |
| 21 | Testing | Hardware and software work properly | DONE |
| 22 | Report writing | Deliver report for comments | (DELAYED) |
| 23 | Report sharpening/Expo | Final delivering | (DONE) |