Scientia Africana, Vol. 13 (No.2), December 2014. Pp336-343
 ISSN 1118 - 1931

VISUAL BASIC APPLICATION IN COMPUTER HARDWARE CONTROL AND DATA ACQUISITION

[1] A. O. Egwali and J. A. Igimoh[2]

[1]*Department of Computer Science, Faculty of Physical Sciences, University of Benin, Benin City, P.M.B. 1154. Nigeria.*

[2]*Department of Electrical Electronic, Faculty of Engineering, University of Benin, Benin City, P.M.B. 1154. Nigeria.*

[1]*E-mail: egwali.annie@yahoo.com*

Received: 06-05-14
Accepted: 16-06-14

ABSTRACT

Speech Recognition Technology enables text acquisition via users' dictation and knowledge gain through system dictation. In this paper the application of speech recognition technology in hardware device control and data acquisition is experimented using Visual Basic and the Speech Application Programming Interface (SAPI) Software Development Kit. To control hardware using Visual Basic, all hardware requests were designed to go through Windows via the printer parallel ports which is accessed and controlled by means of the Printer object. A ULN2003A Relay Driver which contains 7 separate Darlington pairs with common emitters and three modified Transistor-Transistor-Logic circuit (i.e. 74LS365) were used to interface an analog-to-digital converter and the parallel port of the computer. The seven light emitting diodes were driven by the ULN2003A with each pair being rated at 500mA, which can also withstand surges up to 600mA. The proposed speech recognition system enabled data read by the control software to be stored in a file for later use. Also the system was able to reads the system input state via words like "one" which kindle output 1, "four" or "for" for output 4, etc of the different input port to the interfacing hardware device.

INTRODUCTION

There are several categories of speech recognition software (Wiki-Speech, 2014). Many speech recognition systems have only text-to-speech, speech-to-text or both. Researchers have been striving to merge text to speech, speech to text and device control technologies (Rosenfeld, 2000). For some, the original design may come with text-to-speech and later speech recognition may be installed either through an explicit loading of a speech package, or more commonly, through an application which has incorporated speech into it. However, the most common type has dictation using continuous speech, which means speaking words without pauses in between. This is not quite natural, but it is a major goal of speech technology, which involves the development of a natural user interface that incorporates the natural language commands so that instead of using a set of specified commands, with merely saying what is required the computer system takes the appropriate action, thus allowing dictation using continuous speech directly into all Windows application (Cox et al, 2000).

Several approaches have been employed to advance speech recognition technology. IBM developed ViaVoice Gold (Fulton, 2000) which was the first continuous speech product that allows dictation using continuous speech directly into all Windows application. Burileanu et al. (2010) developed a high-quality text-to-speech synthesis system in Romanian language and presents a telecommunication platform based on a client-server architecture and standardized signaling protocols for accessing text information through communication channels. Lindasalwa et al (2010) proposed a voice recognition algorithm using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) techniques. Ibrahim (2010) submitted a technique that incorporated Frequency spectral information to the conventional Mel spectrum base speech recognition process in a Hidden Markov Model (HMM) based recognition approach.

This paper describes three speech technology capabilities of computer system to (i) convert a spoken voice into electronic text (speech recognition), (ii) play back text in a spoken voice (text to speech), and control an electronic device via speech recognition. From programming point of view, these three are independent of each other. Visual Basic, a high level language that is a descendant of BASIC (Beginners all-purpose symbolic instruction code), can be used effectively to develop a computer interface software for speech recognition using the Speech Application Programming Interface (SAPI) of Visual Basic. SAPI also provides an API layer between applications that use speech technology and speech engines. The SAPI Suite contains source code documentation and speech engines designed for developing text-to-speech and speech recognition applications. To enable access to example sample codes for building speech recognition software that enable a program to understand the spoken command of a user and to talk back to the user, the Software Development Kit (SDK) is also utilized and using the SAPI SDK, the design of the text-to-speech and speech recognition application can be done in reasonable amount of time.

Every printer manufacturer develops a printer driver, which implements a set of specified functions on their printer. Application developers are meant to use these set of specified functions to talk to a printer irrespective of the printer model, simplifies application development and applications portability with every printer. In contrast, many application developers write codes to support different speech recognizers, to resolve this multifaceted approach, the speech research group, released different versions of SAPI to specify drivers, like the case of printers, called speech engines (Microsoft Speech, 2012). SAPI is designed to provide an API layer between applications that use speech technology and speech engines. The SAPI Suite contains source code documentation and speech engines designed for developing text-to-speech and speech recognition applications. However, the gaping omission was a COM interface to the runtime so that Visual Basic and other COM - compliant programming environments could use it (Microsoft Speech, 2012).

MATERIALS AND METHODS

The objective of the study is to design hardware for personal computer (PC) control which involves switching on and off devices and data acquisition which involves

Scientia Africana, Vol. 13 (No.2), December 2014. Pp336-343
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

reading the input port periodically and saving the data in a file.

The hardware construction was carried out in the following stages: component selection, bread boarding, soldering, component interconnection, speakers, keyboard and visual Display Unit. For the component selection, three 74LS365 TTL integrated circuits (see figure 1) were used to interface between the ADC, the parallel port and the relay driver. A TTL buffer is used to ensure that no interfacing circuit was required for the parallel port, since the parallel port output is TTL. Supply voltage (V_{cc}) of 5V was used as the 74LS365 could vary between 4.75V and 5.25V. The TTL circuit uses multiple emitters at its input which results in a lower capacitance and higher switching speed.

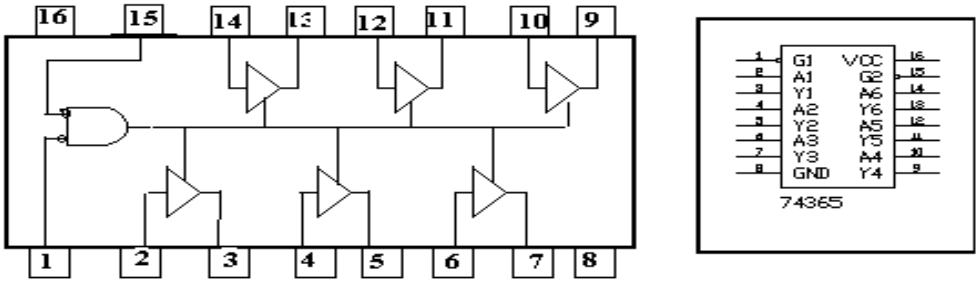

Figure 1: Diagram of 74LS365

ULN2003A that contains 7 separate Darlington pairs with common emitters (see figure 2), each pair rated at 500mA was used to drive the L.E.Ds. In driving the L.E.D., two important policies were taken into consideration. The ULN2003A inputs are compatible with TTL and 5V CMOS so there was no need for special interfacing circuits between them and the Non-inverting buffer. LM7805CP with an output voltage of + or - 4% was used to supply voltage of 5V and since. The CP version of 7805 was used because it is capable of supplying 1A.

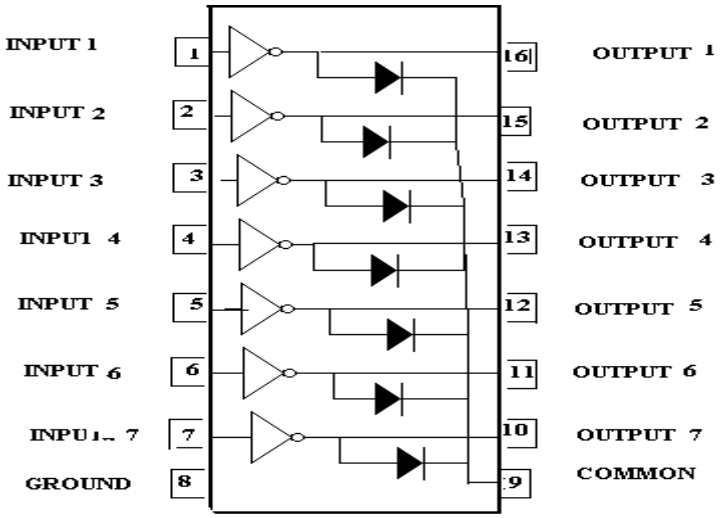

Figure 2: ULN2003A

The hardware requests were made to go through Windows, via the computer parallel port using the Printer object. To control the three different parallel ports which are made up of the standard 8 data lines (information output), 4 control lines (information output) and the 5 status input lines directly, WIN95IO.DLL, a Dynamic Link Library (DLL) was copied to the windows\system directory. The output data is passed from the control software to the parallel port. The data from the parallel port is then passed through the 25way D-type (figure 3) cable to the inputs of the buffer. The outputs of the 74LS365 latch the data. The data 1 to 7 outputs are connected to the relay driver (ULN2003A) where they are inverted and are used to drive L.E.D.'s. The input for data acquisition is connected to the other inputs of the 74LS365. They are buffered and passed to the parallel port through the 25way D-type cable. The data is then read by the control software and stored in a file for later use.

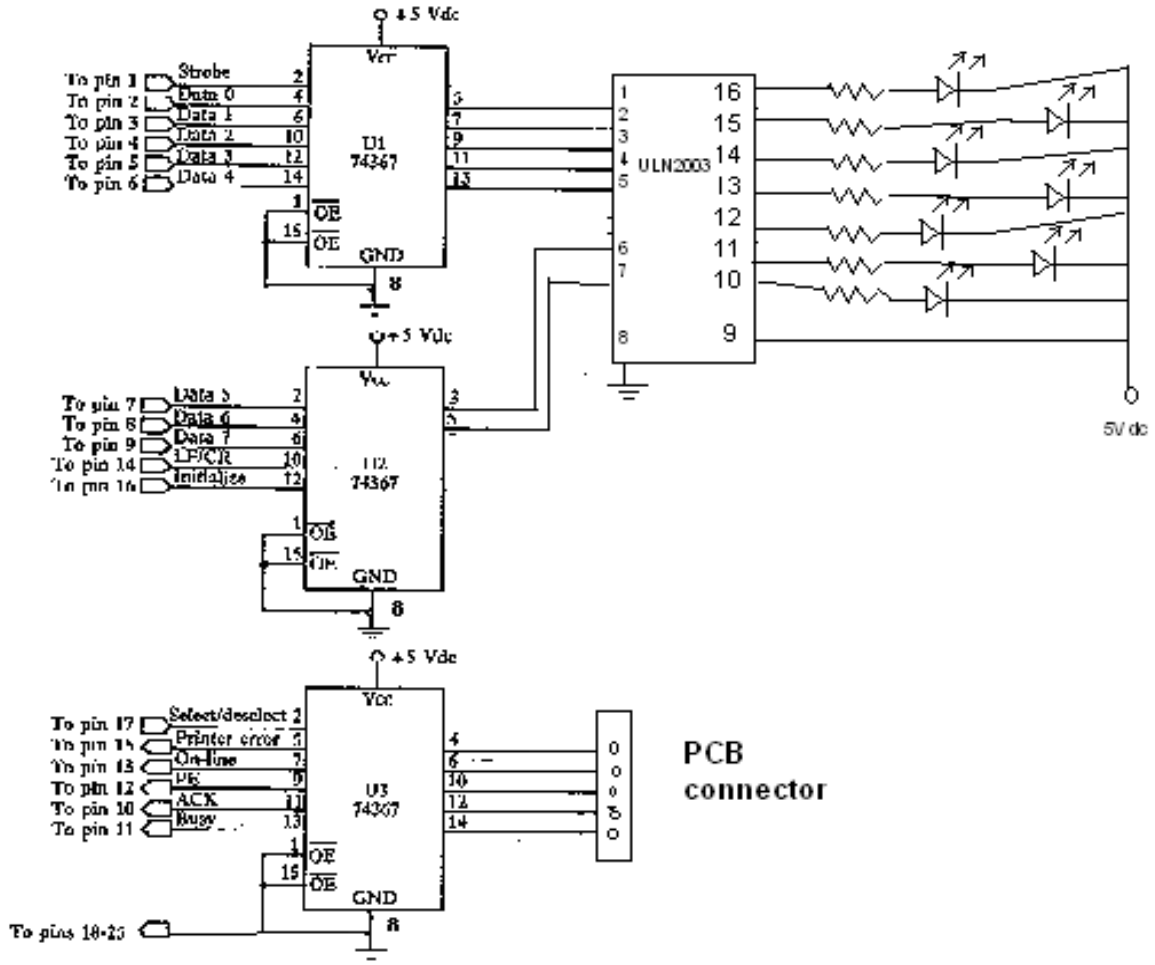

Figure 3: Circuit diagram of Speech Recognition Hardware Control

Next the project was first bread boarded to test the functioning of the circuit. The third phase involved building the basic circuit on one Vero board, which is an insulating board on which copper strips have been bonded, with equally spaced holes drilled along the copper strips.

Scientia Africana, Vol. 13 (No.2), December 2014. Pp336-343
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

The components were assembled on the other side of the board, the components were held in place by pushing their legs through the holes. IC sockets were used for all the IC s so as to ensure easy replacement of IC's in case they get damaged. The IC sockets were soldered to the Vero board first and all other passive components i.e. the resistors and capacitors were soldered next. Three TTL Integrated circuits were used and two CMOS circuits were used. The CMOS IC had to be handled with care since they are easily destroyed by static electricity. An anti-static wrist strap properly earthed was used to hold the CMOS IC before placing them on the sockets. The voltage reference ZR423 was soldered next. A fully screened 25-way D-type male to female cable having 25-way D-type plug at one end and 25-way D-type socket at the other end was used to connect the parallel port to the Vero board.

The socket end was cut off and stripped. The stripped end was soldered to the board. The cover of the container was pierced with 7 pairs of holes. Each pair of holes was for each L.E.D. The legs of each L.E.D. were passed through each hole and all the cathodes soldered together. The other leg was soldered to each of the 7 outputs of the ULN2003A. Next a rectangular hole was made for the power switch and the switch was screwed on. A hole of diameter 10mm was made for the 25D-way printer cable. The power cable was then soldered to the Vero board. The PCB connector was joined end to end so that it has 6 contacts. The legs were now soldered to a small hexagonal shaped veroboard. It was made in that shape so that the cover of the case could still close. The Vero board was now gummed to the case with Super Glue. At the side of the case a rectangular hole was made for the adaptor connector and was glued to the wall of the container. The completed Vero board was placed in a plastic container.

For data acquisition which involves reading the input port periodically and saving the data in a file, Visual Basic programming language was used to control the port directly using the WIN95IO.DLL. To use any functions contained in the DLL, the following functions are declared:

Declare Sub vbOut Lib "WIN95IO.DLL" (ByVal nPort As Integer, ByVal nData As Integer)

Declare Sub vbOutw Lib "WIN95IO.DLL" (ByVal nPort As Integer, ByVal nData As Integer)

Declare Function vbInp Lib "WIN95IO.DLL" (ByVal nPort As Integer) As Integer

Declare Function vbInpw Lib "WIN95IO.DLL" (ByVal nPort As Integer) As Integer

Once the functions are declared, two new commands become available. These are vbInp and vbOut. The vbOut statement enabled the submission of a bit to a port, represented as:

vbOut [port],[number]

The parallel port of the computer used by the Printer is modified for use in the device control and data acquisition. The parallel port is made up of the data lines, control lines and status lines. The ports work with numbers, so each is accessed by its own address which is expressed in hex, binary or decimal as shown in Table 1.

Table 1: Sections of the Parallel port.

Port	Address (Decimal)	Address (Hex)
Data Lines	888	378h
Control Lines	890	37Ah
Status Lines	889	379h

To access the ports, two parameters required are the port address and the value to set the port to. For instance, in the following source code statement:

vbOut [port],[number]

[port] denotes the port address while *[number]* denotes the decimal number to be sent to effect the output of the port.

For the data lines were used for information output and the address of the port is 888, with only 8 output data lines and a maximum value of 255 (i.e. only a maximum bit pattern value of 11111111 can be sent). To set the 8 data lines to 11111111, the value 255 is sent through the parallel port of the printer. Similarly, to set the data lines to 00000000, the value 0 is sent.

For example, setting port to 10110000 is denoted as:

vbOut 888, 11

When a bit pattern is sent to the port everything that was there previously is cleared. The control lines with address port 890, has only 4 outputs, thus the highest decimal representation of the binary bit pattern was 15 (binary 1111). vbInp function was used to read the bit pattern of a port. This is used in the following way:

[variable]=vbInp([port])

To get the current status of the status lines (port 889), the following expression was used:

PortNum%=vbInp(889)

The status lines has a port address of 889 and has 5 input status lines which allow the printer to communicate things such as error, paper out and busy status to the PC. PortNum% contains the decimal representation of the binary bit pattern present at the 5 status lines. vbInp is used to read the status of the output port, and can also be represented as:

PortNum%=vbInp(888)

where PortNum% is set to the current value of the data lines (port 888).

The port address of the control lines is 890. They provide control outputs signals to the printer (such as form feed or initialize) and has only 4 output data lines with a maximum value of 15 (i.e. only a maximum bit pattern value of 1111 can be sent). Only a maximum value of 63 can be sent (i.e. bit pattern value of 111111). When a bit pattern is sent to the port everything that was there previously is cleared.

RESULTS

Testing of the Voltage Regulator

The output voltage was 5V for input voltages from 5.5V to 15V in steps of 0.5V. Test result shows that the voltage regulator produces a constant voltage with varying input voltage.

Testing of Hex Non-Inverting Buffers (74ls365)

A logic probe was placed at each output and a wire was connected from the positive 5V power supply to each input in turn. Each time the logic probe turned on. Next a wire was connected from ground and each time the logic probe was not turned on.

Scientia Africana, Vol. 13 (No.2), December 2014. Pp336-343
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

Testing of Uln2003 (Darlington Array)
A wire was connected from the positive 5V power supply to each input in turn. Each time the corresponding output L.E.D. turned on. Next a wire was drawn from ground and none of the output L.E.D. was turned on.

Testing of Input Port
A logic low was applied to each input and each time the output read with the control software the readings shown in table 2.

Table 5.1 Output Readings from Input to Port

Input Grounded	Digital Output in Binary	Digital output in decimal
All high (no input connected)	0111 1111	127
BUSY pin low	1111 1111	255
ACK pin low	0011 1111	63
PE pin low	0101 1111	95
ON-line pin low	0110 1111	111
Printer Error pin low	0111 0111	119

L.E.D Control and Data Acquisition
The data read by the control software is stored in a file for later use. Some of the dictated text (output data) controlled the LED of the hardware device. The output data is passed from the control software to the parallel port. The outputs of the 74LS365 latch the data. The LED was controlled – turn on and off – by dictating. For hardware L.E.D. control purposes only, the dictated words were restricted to *one* for output 1, *tool* or *two* for output 2, *big* for output 3, *four* or *for* for output 4, *fine* or *five* for output 5, *six* for output 6 and *computer* or *seven* for output 7 using continuous speech directly into a Windows application (text box). The system output the data acquired through the speech of the user to the output port every 500ms to aid in the controlling of the attached device.

DISCUSSION
Many speech recognition systems only have either TTS or SR. For some, the original design may come with TTS and later SR may be installed either through an explicit loading of a speech package, or more commonly, through an application which has incorporated speech into it. However, this paper described three speech technology capabilities that are independent of each other from a programming approach using Visual Basic — the capabilities for a computer system to convert a spoken voice into electronic text (speech recognition), to play back text in a spoken voice (text to speech) and the control of an electronic device via speech recognition. The design and implementation of an electronic device constructed with light emitting diodes depicts the ability of the Computer interface for hardware control and data acquisition using the parallel port of a personal computer.

REFERENCES
Burileanu, D., Negrescu, C. and Surmei, M. (2010). Recent Advances in Romanian Language Text-To-Speech Synthesis. Proceedings of the Romanian Academy, Series A, of The Romanian Academy Volume 11, Number 1/2010, pp. 92–99

Cox, R. V., Kamm, C. A., Rabiner, L. R., Schroeter, J. and Wilpon G J (2000).

'Speech and language processing for next-millennium communications services.' Proceedings of the IEEE 88(8), 1314–1337.

Fulton, S. (2000). *Begin Dictation – Using ViaVoice Gold.* Science and Humanities Press. Pp 27-31.

Ibrahim, P. (2010). "Speech Recognition Using HMM with MFCC- An Analysis Using Frequency Specral Decomposion Technique", Signal & Image Processing An International Journal (SIPIJ), 1: 101-110.

Lindasalwa, M., Mumtaj, B. and Elamvazuthi, I. (2010). "Voice Recognition Algorithms Using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques", Journal Of Computing, 2: 138-143.

Microsoft Speech (2012). SR Engine Vendor Porting Guide SAPI 5.4 - MSDN - Microsoft. Available at: http://msdn.microsoft.com/en-us/library/ee431799(v=vs.85).aspx. Accessed 21 February 13.

Rosenfeld R. (2000). 'Two decades of statistical language modeling: where do we go from here?' Proceedings of the IEEE, Special Issue on Spoken Language Processing 88(8): 1270–1278.

Wiki-Speech (2014). Speech Recognition. Available at: http://en.wikipedia.org/wiki/Speech_recognition. Accessed 17 October 13.