The International Journal of Engineering and Science (IJES)
|| Volume || 6 || Issue || 5 || Pages || PP 08-15 || 2017 ||
ISSN (e): 2319 – 1813 ISSN (p): 2319 – 1805

A Computer Based Artificial Neural Network Controller with Interactive Auditory Guidance

Ahmed Jabbar Abid[1*], Alaa Desher Farhood[1], Maham Kamil Naji[2]
[1] *Middle Technical University, Technical Instrurctors Training Institute, Baghdad, Iraq*
[2] *Middle Technical University, Institute of Techonology , Baghdad, Iraq*

ABSTRACT
The proposed design offers a complete online and offline solution to manage the industrial systems. The designed hardware able to, read analog signals, digital signals, and controls many devices in real time. The heart of the hardware part is microcontroller PIC18F4550 which communicate with a computer via USB. The software part is programmed using Visual C# software to control managed system requires. The system operator can monitor system and diagnostic faults manually or automatically based on artificial neural network. Finally, the system has been simulated and implemented successfully.

Date of Submission: 29 April 2017 Date of Accepted: 08 May 2017

I. INTRODUCTION

Industrial systems managed based on sensor readings which translated by the system; these signals are processed by the system, then controlled all the other actuators, motors, alarms, etc. The sensor's signal could be in the form of digital or analog signals. Digital signal has a voltage level of 0 or 5 V, but analog signals have different voltage ranges like 0-5V, 0-10V or 4-20mA. The proposed system is designed to read eight analog signals, eight digital signals, and controls thirteen devices in real time. All the measured data are displayed on the computer screen and it's possible to run any device automatically or manually.

Automatic mode can be written by the system expert user to manage the entire readings base on the process requirements and the previous experience in the system. Manual mode is should control by the operator, the system will follow the operator orders to run any device, but it still needs a password to log in. A dummy operation has been taken into consideration in manual mode. The operator will be surveillance by the system software for any wrong command. The auditory warning voice played on the computer using a saved message in the C#. These messages can be changed easily according to the requirements. For now, the design has some auditory warnings like if any analog sensor goes below 10% or over 90%. Also, it's notified the operator about any digital sensor status update.

The designed system is automatically predicting and detecting any fault according based on soft implemented artificial neural network (ANN). Weights and biases are measured separately, then used by the designed software. The system designed to save data periodically to shared text file which have a privilege to read and controls remotely by the user.

II. DECISION-MAKING NETWORKS

Human experience through his long journey at any skills could be concluded on his way to analyze its system measurements to detect any fault. So, if we can build such system that making decision based on learning data and can classify and predict any fault. The good news that this system can be built softly and the presented system adopts Single-layer Perceptron Classifiers (SPC). This neural network can be trained to predict and possibly fault according to our digital or analog inputs as a pattern classifier as in Fig. 1 then makes decision.

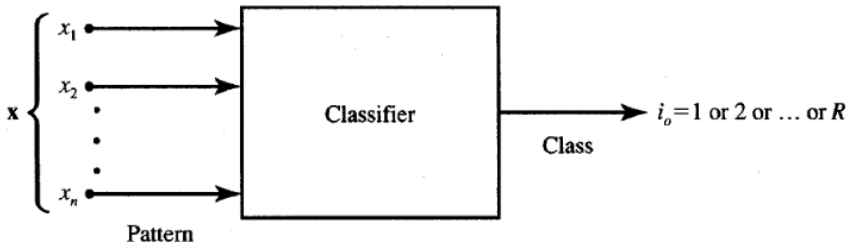

Fig. 1 Pattern classifier

Assume that each such set member consists of real numbers corresponding to measurable results for a given physical situation. Usually, the converted data at the output of the transducer can be compressed while still maintaining the same level of machine performance. The compressed data are called features. The feature extractor at the input of the classifier performs the reduction of dimensionality [1]. Decision-making offers by this system based on a trained artificial neural network programed softly by Visual C# which make training easier and expandable.

III. SYSTEM STRUCTURE

The offered system consists of three parts, as shown in Fig. 2, software, firmware and hardware:

A. Software:

Visual C# is intended to be a simple, modern, general purpose, object-oriented programming language [2]. The computer software used in this design reads and writes 64 bytes from the controller USB buffer periodically via USB:

byte[] DataWrite = new byte[64]; //Buffer USB write
byte[] DataRead = new byte[64]; //Buffer USB Read

The microcontroller firmware load the 64 bytes write buffer periodically or according to any HID interrupt. As a part of the visual C# duties in this design is decision making and this programed based on *Neuralfactor Class* which is a public class.

public class NeuralFactor

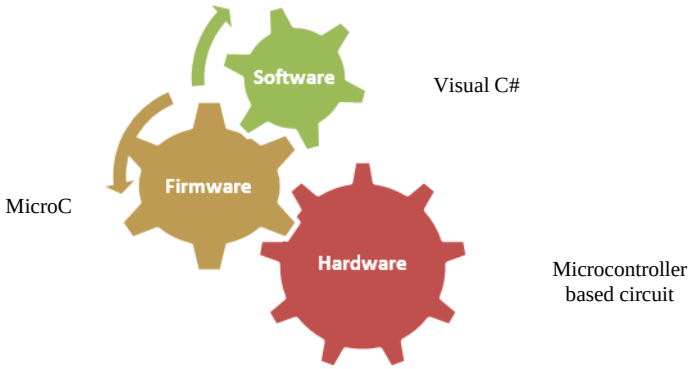

Fig. 2 System Architecture

B. Firmware:

The offered design use PIC18F4550 which is a USB Microcontrollers that load the sensor reading into USB read buffer and write the USB output buffer into relays according to the connection in Table I., where the firmware written by using MicroC software as following:

To read digital signal:
If (readbuff [0] = =1) {PORTB.F0 =1 ;}
Elseif (readbuff [0] = =0) {PORTB.f0 =0 ;}
To write digital signal:
If (portd.f0==1){writebuff[0]=0x01;}
Elseif (portd.f0==0){writebuff[0]=0x00;}
To read analog signal:
Temp_res= ADC_Read(0); // Get ADC0
writebuff[8]= temp_res>>2;// to shift low bits

Table I USB Buffer Data

Buffer	Port	Signal type
Readbuff [0-7]	RB 0-7	Digital Output
Readbuff [8-12]	RC 0-2, 6-7	Digital Output
Writebuff [0-7]	RD 0-7	Digital Input
Writebuff [8-15]	ADC 0-7	Analog Input

C. Hardware

1. Microcontroller circuit

It is required to use a microcontroller with an ability to communicate with a computer via USB. So the selected Microcontroller 18F4550 has this ability also it has USB V2.0 Compliant, low speed (1.5 Mb/s) and full speed (12 Mb/s), supports control, interrupt, isochronous and bulk transfers, supports up to 32 endpoints (16 bidirectional), 1 Kbyte dual access RAM for USB and on-chip USB transceiver with on-chip voltage regulator. The pin connection is shown in Fig. 3.

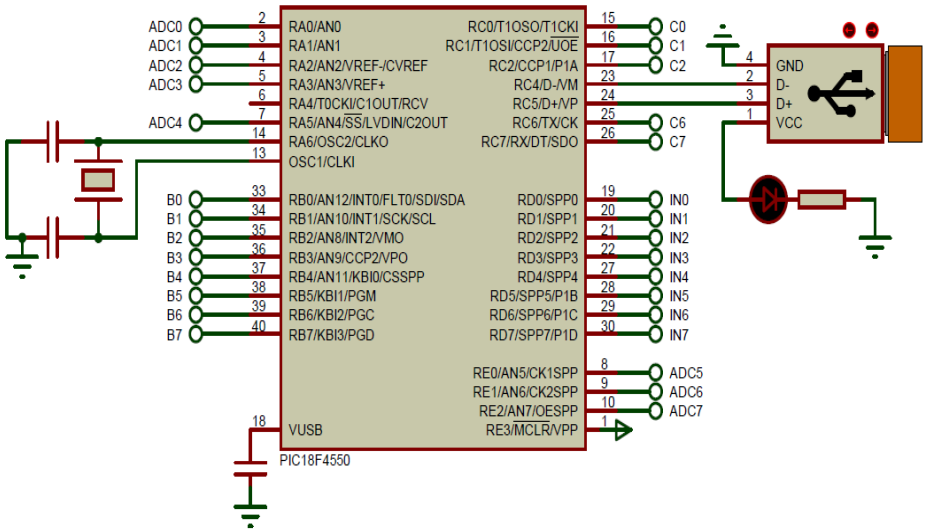

Fig. 3 Microcontroller connection pins

2. Digital input driving stage

This stage buffers an eight digital input signals, as shown in Fig. 4. The received digital signal is optically coupled as a protected stage, then delivered to the microcontroller through port D. Two of a PC847AB have been used to drive the eight input signals [3].

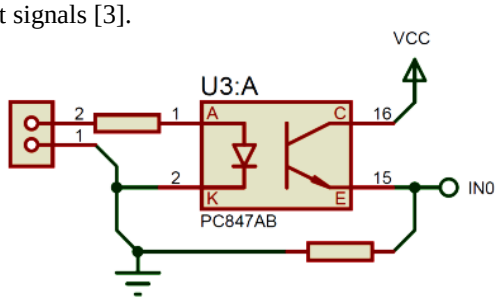

Fig. 4 Digital input driving

3. Analog Input driving stage

This stage is for conditioning, eight analog input signals, four of them in voltage level 0-5V, as shown in Fig. 5a. The analog signal is regulated using Zener diode 5V1 as an overvoltage protection. The other four conditioning circuit is with a voltage level 0-10V, as shown in Fig. 5b which are simply voltage divided then regulated before driving to the microcontroller. Input signals can be designed to carry on the power line as in [4] and [5].

(a) Voltage level (0-5V) **(b) Voltage level (0-10V)**

Fig. 5 ADC Conditioning Circuit

4. Digital outputs driving stage

The design controls eight digital output signals, these signals can control appliances, motors, or any loads through relays. ULN3002A used to drive these relays and protect microcontroller from over current as shown in Fig. 6. The used relay is a 12VDC, coil resistance 275Ω. The current ≈ 12/275=44mA, so the driver sounds safe because its drive up to 500mA.

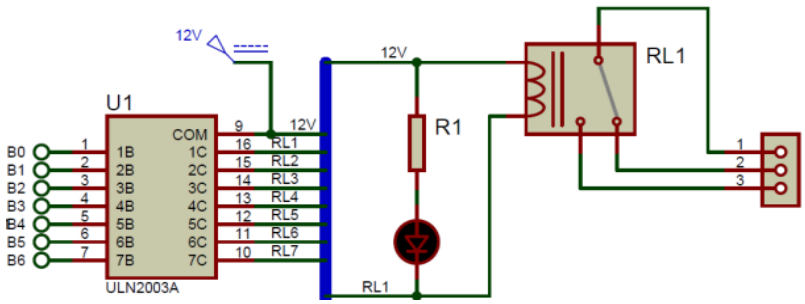

Fig. 6 Relay Driver Circuit

IV. COMPUTER SOFTWARE DUTIES

Visual C# is used to manage the computer side duties. It's programmed to manage the data between the user and the hardware part. These duties include:

D. Initialization

Initialization phase includes establishment of a connection between the computer and the microcontroller via USB port. It shows a *"Data send successful"* message, at the top right corner of the window, in case of the connection is verified and the data sent as in Fig. 7.A, otherwise a *"Device not found"* message will show up as in Fig. 7.B.

A. Connection Establishment Fault **B. Connection Establishment Verified**

Fig. 7 Connection Verification Message

E. Go online with time stamp

The implemented visual C# program saves and reads data from a shared file. This allowed the user to track and control the input/ output signals remotely. Fig. 8 shows the data track file which named "Smart Controller.txt" that shows the status of each input and output signal.

Every single data will save periodically with time stamped, actually every second. A text file, automatically generated by the program to save all the data according to a specific format. This format starts with "*", then followed by ID, then port address, then date and time, and finally the data. It will be as the following format:

*[*ID, A, YYMMDD, HHMMSS, Data#]*

Table II shows the data structure and its specifications.

Table II Data Structure

	Structure	*Specification*
Start		*
ID	DIN	Digital input
	DOUT	Digital output
	ADC	Analog input
Address	1-8	For digital input
	1-13	For digital output
	1-8	For analog output
Date	YYMMDD	Year, Month, Day
Time	HHMMSS	Hour, minute, second
Data	Digital (T or F)	True or false for digital
	Analog (0-255)	Analog level of 8 bits
End		#

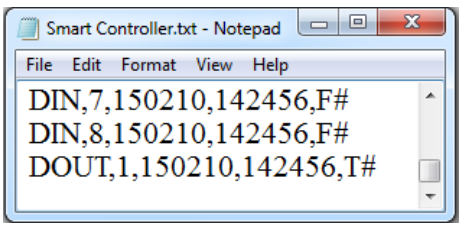

Fig. 8 Generated Text File

F. Fault prediction

Fault prediction or input classification is adopted softly using C# in the design. An example for three digital inputs classified into two classes using perceptron classifier based on Delta Rule Learning. The resultant network weights will have used to make the decision in classifying the inputs as shown in Fig. 9. Many researchers [6] and [7] adopt different techniques in training ANN using C# privilege, but this design does not involve in design but only in adaptation a produced weights and biases matrices to predict outputs which will save manually.

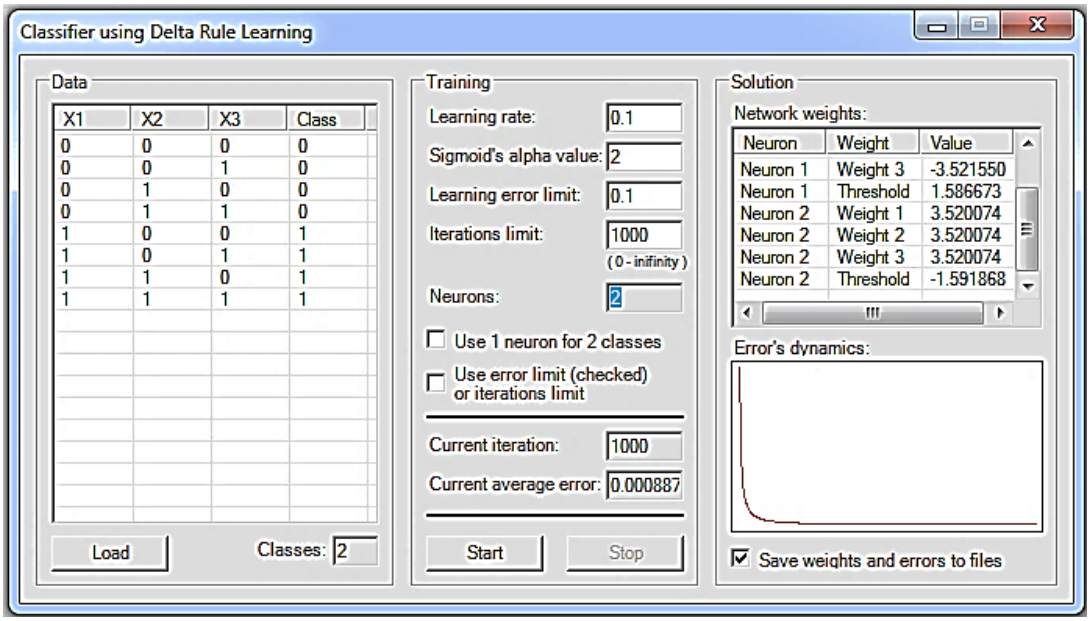

Fig. 9 Classifier using Delta Rule Learning

G. Password Authority

The system has some immunity features like password, the designed software requests a password at the starting or on running some import keys that need to authorize operators only. A pop up window will show as in Fig. 10.

H. Shutdown Procedure

In the industrial application shutdown need a specific procedure. A pop up window will appear as shown in Fig. 11 towarnm the user before shutdown. If the operator presses "YES", the system will load the shutdown program. This program will service this purpose by sequentially shut down the system units, according to a time schedule and this procedure will continue even to disconnect the computer until the system is completely shut down.

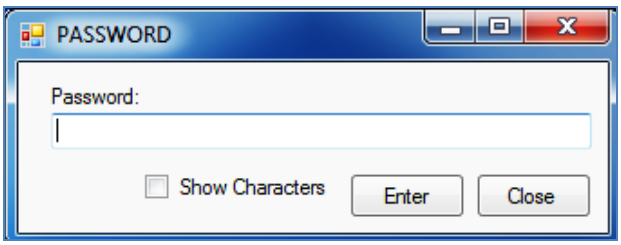

Fig. 10 Password window

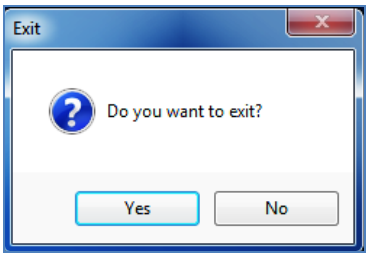

Fig. 11 Shutdown Window

Software duties can be concluded as in Fig. 12 which shows the software flowchart from the starting to shut down.

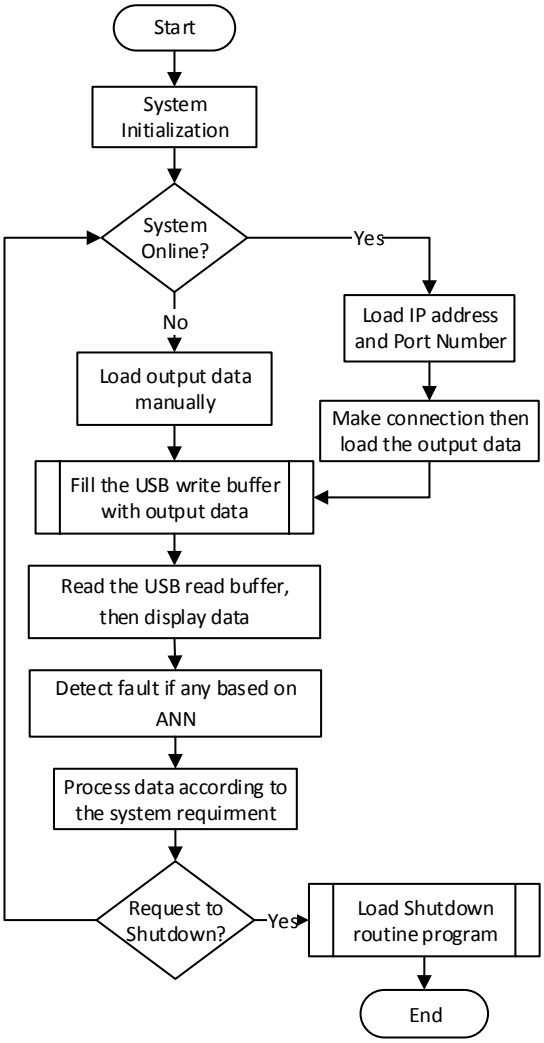

Fig. 12 System Software Duties Flowchart

V. SYSTEM IMPLEMENTATION

System implementation has three parts as mentioned before, hardware, firmware and software. From the hardware point of view, the system implemented and tested as prototype as shown in Fig. 13. After the prototype work, probably a more professional model is implemented using Proteus software as shown in . The proposed hardware model has the following features; compact, low power consumption, reliable, cost efficient and industrious. The firmware is written by using MicroC software which read and write the USB 64 bytes from the input and output signals. Visual C# software is written to be able to communicate with the microcontroller.

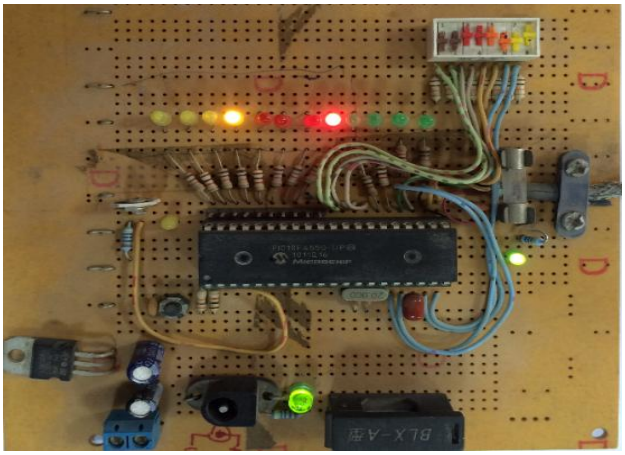

Fig. 13 Hardware prototype photo

Fig. 15 shows the Visual C# window which will used by the user to read digital and analog signals and control all the outputs. In the same figure and from the top to bottom, the first line displays the hardware connectivity, as mention before, it displays a message "*Data send successfully"* or *"Device not found"* according to the connection status. At the top right the window shows the current time and date which will time stamp every reading.

The second line displays the digital input status from DIN 0 to DIN 7. Third line shows the digital output control from D1 to D13. The fourth line shows the eight inputs from ADC0 to ADC7 as a bar display. These bar's colors change according to its values. There is a check box to mute the auditory guidance, otherwise an audio message will play if any input or output signals are changed or the analog signals exceed the upper or lower limits.

VI. CONCLUSIONS

In this article, a smart controller with interactive auditory guidance is presented. The design includes three parts, hardware, firmware and software. The hardware is designed, simulated (using Proteus), implemented and tested perfectly. Firmware is written by MicroC software then compiled to tested in the simulation and prototype.

Main Software is programmed using Visual C# to manage all the activities that issued to the system like online control, decision making to filter the dummy running, fault diagnosis, USB communication with the hardware, auditory guidance and shut down procedure. The implemented system offers a smart, low cost controller that uses a real experience that can manage system remotely or manually.

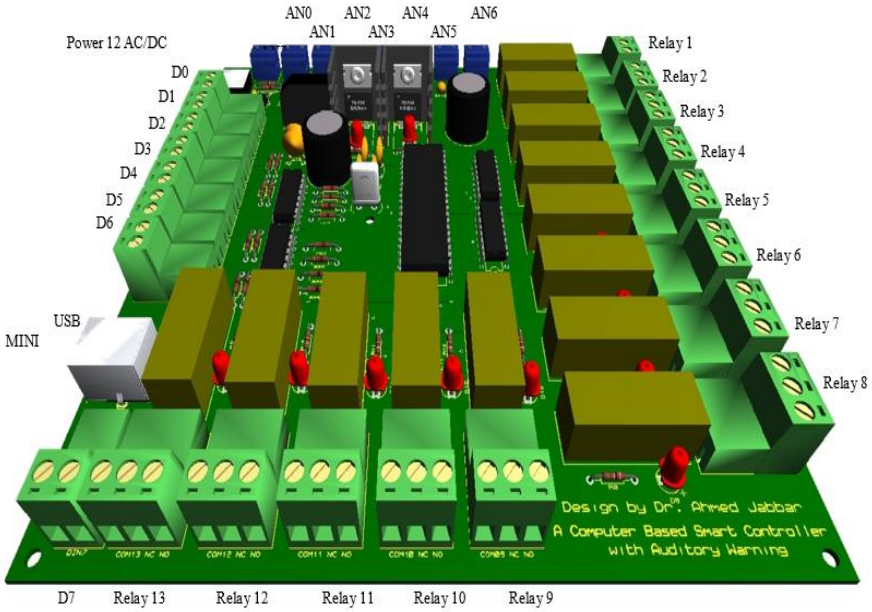

Fig. 14 System 3D view using Proteus

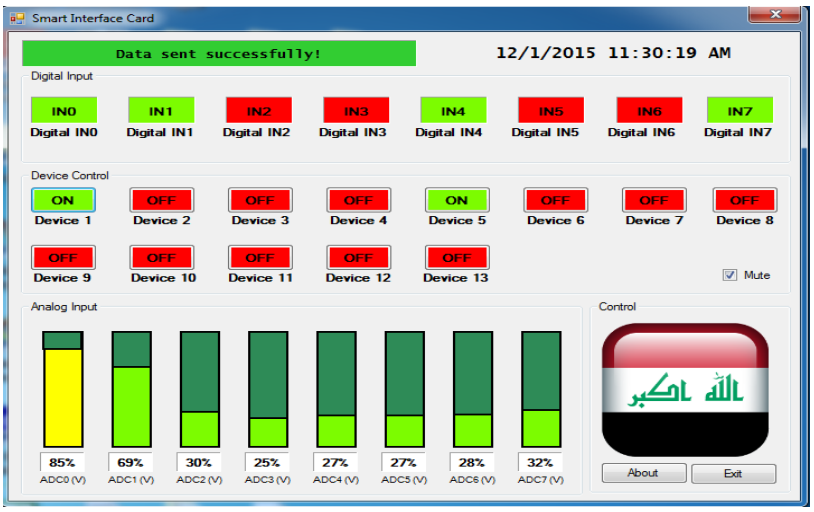

Fig. 15 Software window using Visual C#

REFERENCES

[1]. J. M. ZURADA, Introduction to Artificial Neural Systems, The United States of America: WEST PUBLISHING COMPANY, 1992.
[2]. C# Language Specification, 4th ed. ed., ECMA International, 2012.
[3]. I. Hickman, Analog Circuits Cookbook, Oxford: Newnes, 1999.
[4]. F. Al-Naima, R. Ali, A. Abid, Z. Ghassemlooy and Z. Gao, "A New Power Line Communication Modem Design with Applications to Vast Solar Farm Management," in EPECS 2013, Istanbul, 2013.
[5]. B. Belvedere, M. Bianchi, A. Borghetti, A. C. Nucci and M. Paolone, "A Microcontroller-Based Power Management System for Standalone MicrogridsWith Hybrid Power Supply," IEEE TRANSACTIONS ON SUSTAINABLE ENERGY, vol. 3, no. 3, pp. 422-431, 2012.
[6]. J. Heaton, roduction to Neural Networks for C# (2nd Edition), Heaton Research, 2008.
[7]. R. Tadeusiewicz, R. Chaki and N. Chaki, Exploring Neural Networks with C#, New Work, USA: CRC Press, 2015.