DEVELOPING A SOFTWARE TO DETERMINE THE MICROCONTROLLER SPECIFICATIONS FOR FUZZY LOGIC CONTROL APPLICATIONS *

Bulanık Mantık Kontrol Uygulamalarında Kullanılacak Mikrodenetleyicinin Özelliklerinin Belirlenmesi İçin Bir Yazılım Geliştirilmesi

Abdullah PADAK
Department of Electrical and Electronical Engineering

Murat AKSOY
Department of Electrical and Electronical Engineering

ABSTRACT

In recent years, the Fuzzy Logic Control Systems (FLCSs) have become widespread and their application areas have also extended. There are numerous software developed to use in design of Fuzzy Logic Control Systems. In this study, a Microsoft Windows operating system based, user-friendly software to determine the optimum microcontroller specifications for Fuzzy Logic Control applications is developed. With this software the optimum microcontroller might be chosen. Regarding to performance and cost effectiveness, the best solutions might also be provided. Besides, the microcontroller selection criteria is defined and formulated after studying various applications in different fields. According to this, an algorithm, which forms the fundamentals of the software, is composed. The software, which is developed by “Microsoft Visual Basic” visual programming language, computes the minimum performance and capacity values for the intended application by using the parameters, which are defined by the user and display the optimum microcontroller specifications.

Keywords: Fuzzy Systems, Fuzzy Logic Controllers, Microcontroller for Fuzzy Logic Control, Software for Fuzzy Logic Control Systems.

ÖZET

Bulanık Mantık Kontrol Sistemleri (BMKS) kullanımı son yıllarda iyice yaygınlaşmış ve uygulama alanı da genişlemiştir. BMKS tasarım çalışmalarında kullanılan mikrodenetleyicinin seçilmesi için birçok yazılım geliştirilmiştir. Bu çalışmada, BMKS uygulamalarında kullanılmak üzere birçok yazılım geliştirilmiştir.


Anahtar Kelimeler: Bulanık Mantık, Bulanık Mantık Kontrol Sistemi, Mikrodenetleyici, Mikroişlemci.

* MSc. Thesis-Yüksek Lisans Tezi
Introduction

Fuzzy Logic Control (FLC) is an up-to-date control method which is mostly utilized in nonlinear applications and for the systems which are not easy to define and model mathematically and suitable to describe linguistically. For a long time it has been preferred because it has severe advantages in some systems. Each input/output variable can be represented linguistically. And all the operations between these variables can be done linguistically. This provides the designers flexibility; everything that can be expressed linguistically can be represented in FLC.

Fuzzy Logic(FL) is a departure from classical two-valued sets and logic, that uses "soft" linguistic (e.g. large, hot, tall) system variables and a continuous range of truth values in the interval [0,1], rather than strict binary (True or False) decisions and assignments.

"Fuzzy Logic Control" (FLC) gained an increasing popularity specifically for especially various control applications in recent years. It even started to be used for the control applications of home electronics. The main reason for this is the increasing importance of energy class evaluation for all kinds of electronic goods both industrial and home-use applications. Regarding to increasing complexity and required better control quality required better control characteristics and energy saving, thrifty control systems. Even for some applications especially for the ones which are difficult to model mathematically and for nonlinear systems the FLC is the remedy for implementing the easiest solution.

FL is a new theory extending our capabilities in modeling uncertainty (Larsen, 1997). Uncertainty is one of the application areas of FL because uncertainty can best be modeled by using linguistic variables. Some adaptive control techniques can be productive for this subject. The goal of the adaptive controller is to provide stable control of systems with significant uncertainty (Spooner et al., 2002).

Fuzzy set theory provides a major newer paradigm in modeling and reasoning with uncertainty. Though there were several forerunners in science and philosophy, in particular in the areas of multi-valued logics and vague concepts, Lotfi A. Zadeh, a professor at University of California at Berkeley was the first to propose a theory of fuzzy sets and an associated logic, namely fuzzy logic (Larsen, 1997).

Materials and Methods

The main parameters for a basic microcontroller (µC) are processing speed and memory volume. For a FLC the parameters which are effective on these parameters are: Number of knowledge base, Number of fuzzy inputs, Number of fuzzy outputs, Input resolution, Number of fuzzy rules, Fuzzy inference speed, Number of shared rules, Number of parallel processors. These fuzzy parameters have been selected after several fuzzy logic systems had been examined.

Some other software particularly prepared for fuzzy logic applications have also been investigated and common features of these software have been listed.
Mostly they are user-friendly and they have a graphical user interface. The proposed software also has a very useful graphical interface. To provide this a visual programming language has been used.

**Finds and Discussion**

Fuzzy logic is a superset of conventional (Boolean) logic that has been extended to handle the concept of partial truth values between "completely true" and "completely false". It was introduced by Dr. Lotfi Zadeh of UC/Berkeley in the 1960's as a means to model the uncertainty of natural language (Web1). The truth of a logical expression in fuzzy logic is a number in the interval \([0,1]\). Fuzzy logic has emerged as a profitable tool for the control of complex industrial processes and systems. It is used for processes that have no simple mathematical model, for highly non-linear processes, or where the processing of linguistically formulated knowledge is to be performed. The controllers based on this mathematical approach are known as fuzzy controllers (Cirstea et al., 2002).

Fuzzy logic has the advantage of modeling complex, nonlinear problems linguistically rather than mathematically and using natural language processing (computing with words). The use of fuzzy logic requires, however, the knowledge of a human expert to create an algorithm that mimics his/her expertise and thinking. Also, studying the stability of a fuzzy system is a demanding task (Ibrahim, 2004).

Fuzzy Logic has been applied to problems that are either difficult to face mathematically or applications where the use of Fuzzy Logic provides improved performance and/or simpler implementations. One of its main advantages lies in the fact that it offers methods to control non-linear plants, known difficult to model (Dualibe et al., 2003).

The use of fuzzy logic is relatively simple as compared to conventional system even for the engineers who don't have a control theory background. It was developed by Lotfi Zadeh. Fuzzy Logic tries to emulate some of the properties that human beings use in their control and decision processes (Jackson, 1994).

The most widespread use of fuzzy logic today is in fuzzy control applications. You can use fuzzy logic to make your air conditioner cool your room. Or you can design a subway system to use fuzzy logic to control the braking system for smooth and accurate stops. A control system is a closed-loop system that typically controls a machine to achieve a particular desired response, given a number of environmental inputs. A fuzzy control system is a closed-loop system that uses the process of fuzzification to generate fuzzy inputs to an inference engine, which is a knowledge base of actions to take. The inverse process, called defuzzification, is also used in a fuzzy control system to create crisp, real values to apply to the machine or process under control. Today fuzzy controllers have been used to control many machines, including washing machines and camcorders (Rao, 1995).
Fuzzy Logic Controller

Figure 1. Block diagram of a typical fuzzy logic controller.

Figure 1. shows the block diagram of a typical fuzzy logic controller (FLC). There are four principal elements to a fuzzy logic controller:
- Fuzzification module (fuzzifier).
- Inference engine.
- Defuzzification module (defuzzifier).
- Knowledge base.

Automatic changes in the design parameters of any of these elements creates an adaptive fuzzy controller. Fuzzy control systems with fixed parameters are non-adaptive. In this thesis non-adaptive fuzzy controllers are studied. Other non-fuzzy elements which are also part of the control system include the sensors, the analogue-digital converters, the digital–analogue converters and the normalization circuits.

Architecture of a General Purpose Microcontroller

The study for developing a software, on selecting microcontroller by determining some basic specifications of the microcontroller such as processing
speed, program memory is aimed to describe a general purpose µC. Except for
giving a multi processing option the program has been prepared at this point of
view. But however we can not neglect that there are many different types of
specialized µC developed specially for FLCS. Though this is not in the scope of
this developed software general information will be as well given about these
hardware.

Figure 2. Block diagram of a simple microcontroller
(Bannatyne et al., 1997).

A microcontroller is a single integrated circuit that at least contains the
necessary elements of a complete computer system: CPU, memory, a clock
oscillator, and input & output. Microcontrollers commonly contain additional
peripheral modules, such as serial and timer units. An example of a simple
microcontroller is shown in Figure 2.

The CPU fetches instructions from program memory via the address and
data bus. Instructions are fetched from successive memory locations until the
execution of a branch or jump instruction. Once fetched, an instruction waits in the
instruction register or pipe unit until the CPU control logic is ready to decode it. An
instruction is decoded by control logic to produce control signals. These control
signals are fed into the execution unit to produce micro-operations that perform the
function of the instruction. The execution unit contains:

1) A set of registers (programmer’s model, temporary registers, address,
data & instruction buffers),

2) Functional units, such as an ALU and shifter, and
3) Internal busses to connect the registers and functional units (Bannatyne et al., 1997).

Memory Demand

Two types of memory demand will be discussed. Random Access Memory, which called commonly as RAM and Reading Only Memory which is labeled as ROM.

ROM will keep the data which is one time programmed at the beginning and only used for reading the pre-stored data later while system is being used. Our knowledge base will be kept in this memory. So this will be the largest memory in our system. Since neuro-fuzzy systems are not included in the scope of this thesis. We don’t need to store any extra data permanently while the program is running on our system. We will take into consideration only a memory of ROM.

RAM will have less volume in the FLCS as compared to ROM. All the data will not be required to be kept at the same time in RAM. Only the data which is under operation will be temporarily kept in this memory. So this volume of this memory will very little. Only in some extreme situation the amount of this memory demand will increase significantly.

This parameter determines the memory demand for the application. Knowledge base keeps all the required data for the application. It consist of Input membership functions, output membership functions and fuzzy rule base, number of rules. Resolution should also be taken into account since it is effective on memory demand (Eichfeld et al., 1995).

\[
M = R_{s} \times \left[ \sum_{k=1}^{KB} \left[ 256 \times Ni + (Ni + Nr) \times No \right] + 264 \times No \right]
\]

Here \( R_{s} \) is the resolution, \( KB \) is the number of Knowledge Base, \( Ni \) is Number of input, \( Nr \) is number of rules, \( No \) is number of output. And \( M \) is the memory demand for the proposed microcontroller.

Performance Demand

Since the then-part chip can execute the consequent inference including de-fuzzification every 4 clock cycles and the clock frequency is \( F_{c} \) MHz, the inference speed of the then-part chip is \( F_{c}/4 \) MFLIPS (Fuzzy Logic Inference Per second). While, the inference speed of the if-part chip is dependent on the numbers of inputs, outputs and rules. First, let us consider the dependence on the numbers of inputs and outputs. When the number of inputs and outputs are \( Ni \) and \( No \) respectively, \( (Ni \times No) \) clock cycles are needed at least for the process. So, the inference speed is \( F_{c}/(Ni \times No) \) MFLIPS. Next, let us consider the dependence on the number of rules. Since each processing element executes one antecedent per one clock cycle, \( F_{c} \) (clock frequency) \( \times No \) (the number of processing elements) \( M \) antecedents can be executed per second. So, since \( N \times No \times Nr \) antecedents are processed in case of \( N \)-input, \( No \)-output and \( Nr \)-rule, the inference speed is \( F_{c} \times No \times Nr \)
Therefore, the inference speed (Is) of the system can be expressed as follows (Sasaki et al., 1991):

\[ I_s = \min \left\{ \left( F_c, \frac{F_c}{(N_i \cdot N_o)}, \frac{F_c \cdot N_p}{(N_i \cdot N_o \cdot N_r)} \right) \right\} \text{MFLIP} \]

Here we can say Is (Inference speed) can be equal to \( F_c, \frac{F_c}{(N_i \cdot N_o)} \) or \( \frac{F_c \cdot N_p}{(N_i \cdot N_o \cdot N_r)} \) but for as per the minimum law we must determine the minimum value and take it into consideration.

If we want to calculate the maximum clock frequency of microcontroller for the given application then equation can be reorganized as:

\[ F_c = \max \left\{ I_s, \frac{I_s \cdot (N_i \cdot N_o)}{I_s \cdot (N_i \cdot N_o \cdot N_r)} \right\} \]

Here \( F_c \) denotes maximum required µC speed in MHz for our application. Since this formula will be used in our software to compute the required processing speed we need to take it as the worst case that we can meet for an ordinary general purpose µC. If you are planning to use a special purpose µC developed for Fuzzy Logic Control Systems with special instructions set this value might change specifically.

Developing the Software

The software development has been realized in three phases:
1. Preparing the flowchart of the software.
2. Preparing the program algorithm
3. Writing the program codes in Visual Basic (VB) programming language.

The Flowchart of the Software

In the figure below the flowchart of the software can be seen. Preparing the flowchart is the basic step for software development. The operations which are implemented in the developed software are defined graphically. Loops can be seen easily since they are drawn graphically.

We can summarize the operations shown in the flowchart as follows:
1. Ask for required variables to be entered.
2. Examine these variables by comparing the given reference values.
3. If the entered variables are within the range of reference values they are then assigned and stored.
4. Execute the calculations as per the given equations.
5. Display the results.
Figure 3. Flowchart of the developed software.
Results and Conclusion

Fuzzy logic control systems can be categorized by considering the parameters of the system. As the values for each parameter gets higher the complexity of the system increases. As the complexity of the system increases the memory demand and the performance demand increase.

Most of the applications of fuzzy logic are digital applications, digital embedded applications. So specialized or general purpose microcontrollers are used. Microcontrollers are various and the best performance and the cost effective choice should be done by the designer. Designers can also utilize this study for the hardware selection if they don’t prefer to use all embedded system, or prefer to use a microprocessor instead.

In this paper it is aimed to propose user-friendly software. It is developed by means of a high level visual programming language, Microsoft Visual Basic 6.0. It is considered that, this programming language is the best choice for developing such software with a visual interface. All the parameters may be entered in the same window, all the warnings or information messages even the results can be displayed in the same window. This provides the ability of simply viewing all the data in the same window. This is useful and practical.

The numerical examples prove that certain parameters have certain effects on the amount of program memory and required processor speed. According to the given parameters values, two basic microcontroller selection criteria, namely memory and performance demands, are computed and monitored.
The proposed software is prepared only for the purpose of selecting the proper microcontroller for FLC applications. This software can be as well extended by adding some more solutions for FL.

References