SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

TEMPERATURE CONTROL USING FUZZY LOGIC

By Lionel Picandet

INTRODUCTION

Fuzzy logic may be considered as an assortment of decision making techniques. In many applications like process control, the algorithm's outcome is ruled by a number of key decisions which are made in the algorithm. Defining the best decision requires extensive knowledge of the system. When experience or understanding of the problem is not available, optimising the algorithm becomes very difficult. This is the reason why fuzzy logic is useful.

We can split the problem into a discrete number of possible decisions by associating fuzzy logic membership functions with each input and output. The accuracy of the output depends on how many membership functions we define and how many rules we implement. The outcome is that a user without know-how or an extensive understanding can solve the problem.

This application note describes the use of fuzzy logic to create a temperature controller suitable for home appliance needs. The example uses the ST6225 Microcontroller with the ST6 fuzzyTECH Explorer Edition fuzzy logic development program. Practical steps in the evaluation are demonstrated with the ST6 Starter Kit as the evaluation vehicle.

1 TEMPERATURE CONTROL SPECIFICATIONS

1.1 Application Overview

Figure 1 : Application diagram

$$\theta = Tref - Trad$$
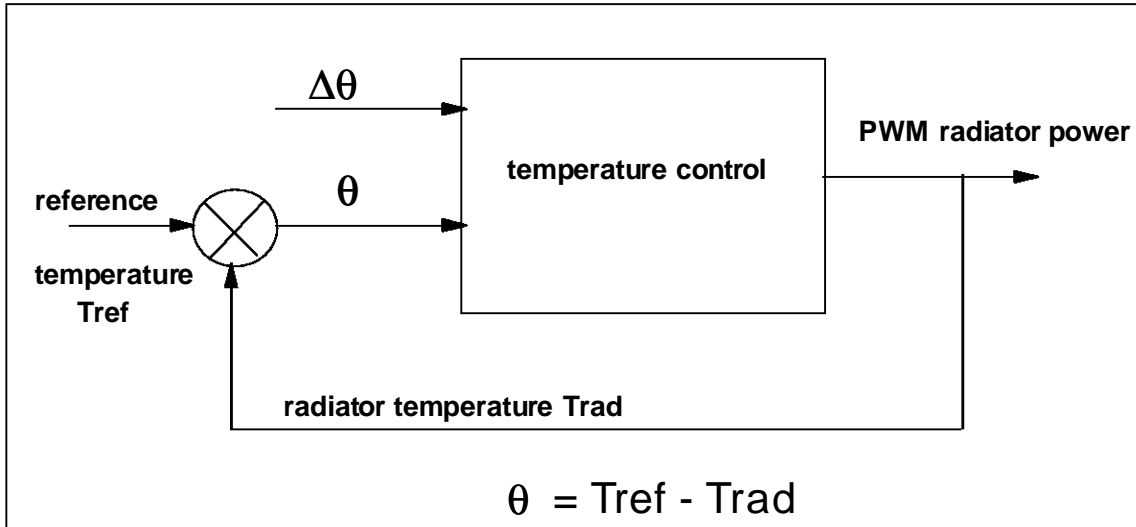
In this application we never refer to absolute temperatures but to difference in temperature. The application shown in this note aims at designing a fuzzy application, so accurate temperature is not important.

The control temperature application consists of regulating radiator power according to a reference temperature. Due to the fact that this is a tutorial example to get started with fuzzy logic, this application does not allow the choice of the regulated temperature. This choice can be developed as an exercise for the reader.

During the initialization phase the ambient temperature is stored in the system and an offset is added. The result becomes the reference temperature. For this reason it is recommended that the resistor is cold before restarting the application.

The system takes into account as inputs the difference between the current temperature and the reference temperature. Another input takes into account the variation of the difference.

1.2 Hardware resources

To realize the temperature control we need the following resources:

- a resistor to simulate the heating source
- a thermistor to measure the temperature of the heating source
- LEDs to display the temperature regulation and the PWM value
- a timer to realize a PWM value to trigger the resistor power
- an analog to digital convertor to connect the thermistor

We have realized the application using the ST6225 microcontroller which suits the temperature control application well. The ST6225 has a timer with an output to trigger the

**SGS-THOMSON**
MICROELECTRONICS

resistor, an analog port to connect the thermistor and logic ports to drive LEDs. It also provides sufficient memory to design the software control.

The temperature control application is implemented with the ST622x Starter Kit which provides all the hardware resources (resistor, thermistor, LEDs) in one easy to use package.

As we deal with converted values instead of temperature, it is necessary to know the transfer characteristics of the thermistor to regulate the temperature in a linear part. In the application the thermistor is connected to a bridge network. The figure below displays the transfer characteristic of the temperature which depends on the converted value of the thermistor bridge.

We use the negative slope of the thermistor transfer function, so when the temperature rises, the converted value decreases.

Figure 2 : Thermistor transfer function

transfer characteristic of the temperature

T = f (code value)
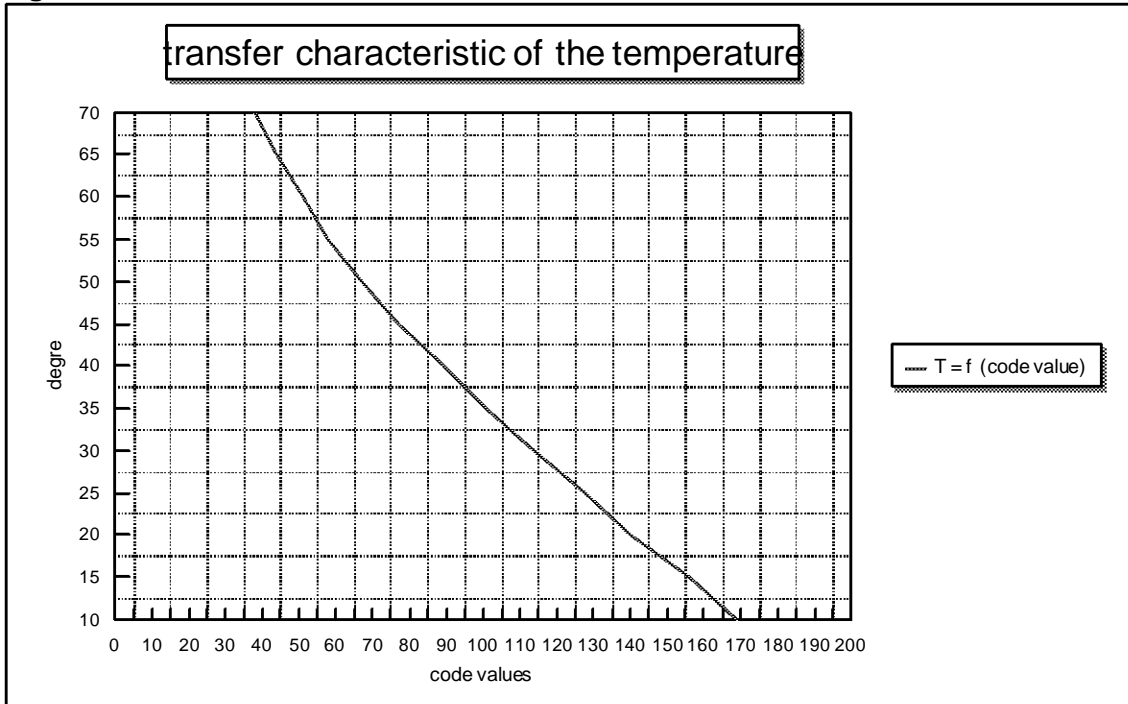
We consider that the ambient temperature is around 25° C. So we can see that we have a linear part up to about 45° C.

SGS-THOMSON
MICROELECTRONICS

1.3 Software and hardware constraints

The ST622x microcontroller has no direct PWM timer function, so the software must take into account the update of the PWM timer. Due to the fact that the fuzzy algorithm takes about 20 ms for two inputs, this delay time influences the granularity of the PWM value. Because the application is an infinite loop and there are several software tasks to do in addition to the fuzzy calculation, we take the maximum counting time for the timer and we consider that the granularity of the PWM timer is equal to the maximum counting time of the timer.

We consider that a granularity of 5% is a good resolution for the PWM timer. As the maximum counting time of the timer is nearly 50 ms, the period of the PWM timer is about 1s. Because the power to the radiator is to be displayed by switching on or off a LED (LED and resistor are directly connected to the output of the timer) and that this LED must visualize the behavior of the application, a one second period is sufficient.
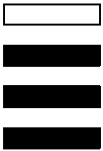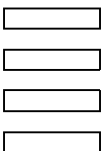
The first LED switches off when the delta is < 3° C
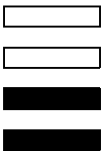
The second LED switches off when the delta is equal to 0° C
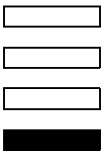
The first LED switches off when the delta is < 0° C

The first LED switches off when the delta is < 3° C

We display the difference between the current and the reference temperature by switching LEDs on or off. When the current temperature is equal to the reference two LEDs are switched off and two LEDs are switched on. At reset all LEDS are switched on, indicating that the current temperature is lower than the reference.

SGS-THOMSON
MICROELECTRONICS

Use of the ST6 fuzzyTECH tool's debugging and optimizing mode

It is necessary to define how to use ST6 fuzzyTECH tool to design the application. The use of a serial link is a good solution to get started with the fuzzy design as this allows direct interaction with the ST6 fuzzyTECH tool, however some constraints need to be observed:

Due to the fact there is a communication link between the PC and the Starter Kit, the transmission must not be interrupted. As we use the timer to realize PWM and that change of the output state can occur at any time, use of the timer in interrupt mode is prohibited. So the timer is used in polling mode.

Another constraint of the serial link is that the serial link is divided into two parts: communication toward the PC and answer from the PC. The time response of the second part depends highly on the PC frequency and memory capacity and it depends also on the number of opened windows in the ST6 fuzzyTECH tool. The time response can take more time than the timer takes to count to the maximum counting time, which can generate an error condition relative to the PWM value. The solution to solve this problem is to consider that a loop is the granularity of the PWM value. The difference between prototyping (use of this method) and the final solution is that the PWM time period is different. We can also consider that the granularity of the PWM value is greater in prototyping than the granularity in final solution.

Nevertheless we give you the worst transmission time that can occur in the application using communication between the PC and the ST6 board. The worst case occurs when we use the serial link debug mode. As we have two input fuzzy variables the number of characters transmitted to the PC is 10. If you use a communication speed of 19200 baud for the serial link, it takes about 5 ms to transmit the 2 input variables. We can sum up this transmission time in the following table (values are approximate):

Communication Baud rate	Typ. Time ms
19200	5
9600	10
4800	20
2400	40
1200	80
600	160
300	320
110	910

1.4 Software development

Before designing the fuzzy system it is necessary to design and validate the software treatment without the fuzzy module. In this way we do not mix potential problems. The software components to be tested are:

SGS-THOMSON
MICROELECTRONICS

- initialisation of the PWM timer and the update of the PWM value
- initialisation of the A/D converter and the conversion
- initialisation of LEDs and the display of PWM and reference.
- scale settings of the linguistic variables

PWM Timer

The ST622X microcontroller has no direct PWM timer function, so Timer 1 is used to achieve this function. During a PWM period it is necessary to split the period in two parts: the first part associated to the one state and the second part associated to zero state. As the PWM timer period is one second and Timer 1 period is 50 ms, the PWM count takes 20 Timer 1 counts; so we need three variables to manage the PWM timer: the PWM period, the low PWM count, the high PWM count such that:

PWM period = low PWM count + high PWM count.

A/D Converter

As the value of the temperature can fluctuate, and according to the accuracy of the A/D converter, it is necessary to filter the converter value. We make 2 conversions and clear the less significant bit of the converted value of each to reduce the effect of any fluctuations. We then compare the 2 converted values; we repeat the conversions as long as the two converted values are different.

Variables Scaling

The ST6 fuzzyTECH tool does not take into account signed values, so it is necessary to convert these values to the interval [0,255]. Using signed values, the zero value is located at the value 127.

Integration

It is necessary to test these components individually before designing the fuzzy application.

SGS-THOMSON
MICROELECTRONICS

2 FUZZY LOGIC DESIGN

2.1 FIRST STEP
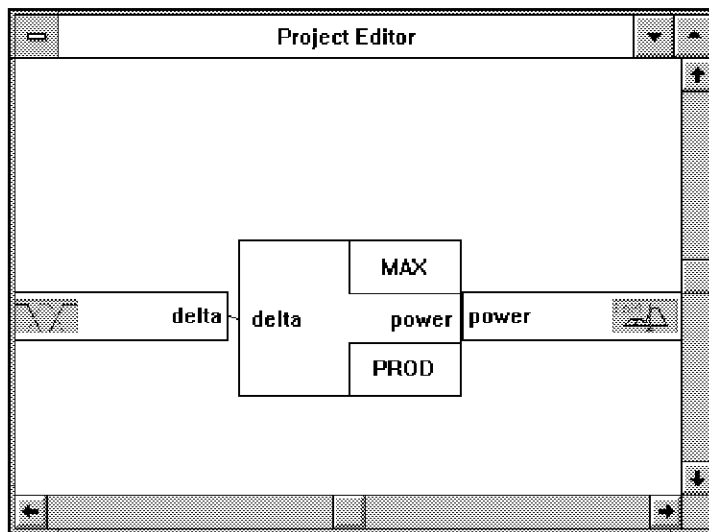
Definition of the variables of the system

To get started with a fuzzy logic design it is necessary to know what you want to realize and to be able to explain how it works. Here it is a temperature control. The goal is to heat a resistor by passing a current through it until its temperature reaches the reference temperature. It is not necessary to have know how about the application to design it with fuzzy logic. It just requires everyday language and common sense. So we can say that:

- IF we are FAR from the reference temperature THEN we need to apply a BIG power to the resistor
- IF the current temperature is CLOSE TO the reference THEN we need to apply a SMALL power
- IF we are EQUAL to the reference THEN we need to apply NO power.

Names like REFERENCE TEMPERATURE and CURRENT TEMPERATURE are the inputs of the application and Power is the output of the system. The actions described above represents the algorithm of the system.

As the reference temperature is always the same, we consider that the difference (delta) between the current temperature and the reference temperature is the input of the system. The input delta and the output power are called the linguistic variables of the system.

Using the ST6 fuzzyTECH tool you can display the application as shown in the figure below:

Once you have defined the linguistic variables of the system, the second step is to define the set of terms that represents each linguistic variable. For the power output we have partially defined these terms before;
The set of terms for the linguistic variable power is [ZERO,SMALL,MEDIUM,BIG].

SGS-THOMSON
MICROELECTRONICS

For the delta input as we consider the difference between the current temperature and the reference temperature, it is important to define negative and positive difference, so the set of terms for the linguistic variable delta is [NEGBIG,NEGSMALL,ZERO,POSSMALL,POSBIG].

This decomposition is not the only possible solution. Each designer can have his own opinion about the system.

Then according to the scale of each linguistic variable and physical representation it is useful to define a typical value for every term, then minimum and maximum limits. As we consider fuzzy sets, there must not be any discontinuity between terms, so terms must overlap each other.

Due to the fact that certain debug modes use code values [0,255], it is necessary to convert shell values into code values.
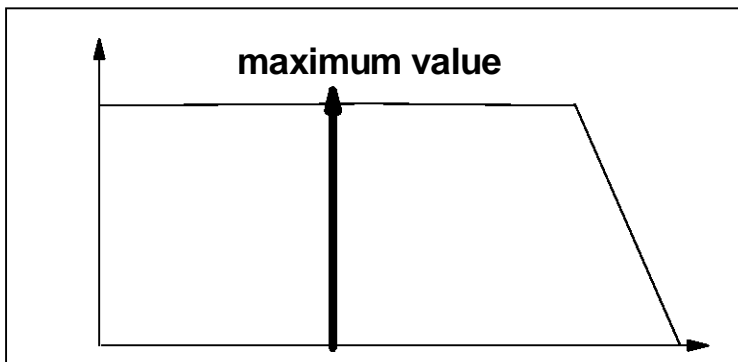
■ **PWM value**

Value	minimum value	typical value	maximum value
zero	0% //0	0% //0	20% //48
small	10% //24	30% //72	50% //120
medium	40% //96	60% //144	80% //192
big	70% //168	90% //240	100%

Percent is the shell unit; Values take part of the interval [0,100%] which corresponds to [0,255] in code values. In the array each item is composed of two values: the shell value and the code value.

NOTE: Using the Center of Maximum defuzzification method needs to put the maximum value for extreme terms (for example BIG) in the middle of the null slope of the membership function. (See the figure below)

Figure 3 : Center of maximum position

■ delta = current temperature - reference temperature

Value	minimum value	typical value	maximum value
negbig	-32 // 87	-5 // 107	-2 // 117
negsmall	-5 // 107	-2 // 117	0 // 123
zero	-1 // 122	0 // 127	1 // 132
possmall	0 // 131	2 // 137	5 // 147
posbig	2 // 137	5 // 147	32 // 167

The shell unit is degrees; we just measure the gap between the current temperature and the reference temperature. Values take part of the interval [-32, +32] which corresponds to [0,255] in code values. In the array each item is composed of two values: the shell value and the code value.

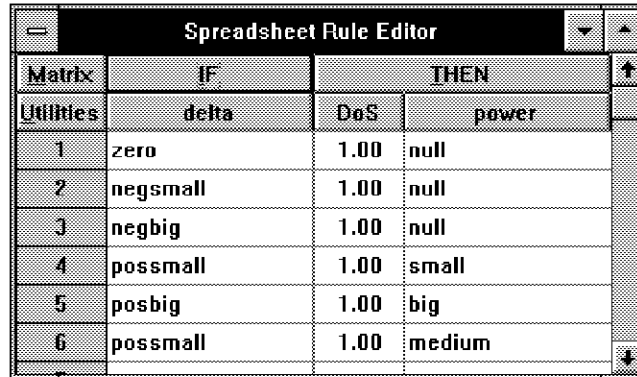Using fuzzyTECH, the linguistic variables and their terms are defined as shown in the figure below:
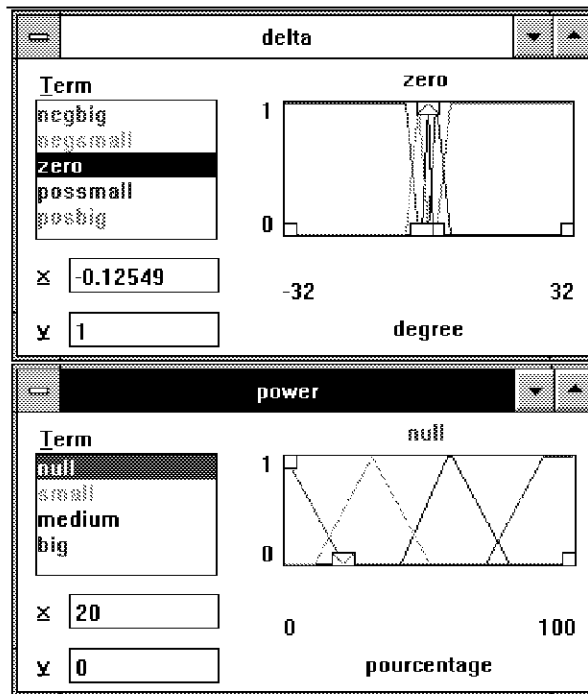
The terms are represented as linear shapes called Membership Functions (MBF).

Algorithm of the system

Once you have entirely defined the linguistic variables of the system you have to describe the rules of the system.

Matrix	IF		THEN
Utilities	delta	DoS	power
1	zero	1.00	null
2	negsmall	1.00	null
3	negbig	1.00	null
4	possmall	1.00	small
5	posbig	1.00	big
6	possmall	1.00	medium

Spreadsheet Rule Editor

You see in the rule editor that for the delta belonging to the POSSMALL term there are two rules defined. The reason is that a SMALL power is not enough to reach the reference.

Prototyping the application

As soon as the application has been designed with the ST6 fuzzyTECH tool, it is interesting to see practical result on the ST6 board.

To prototype the application we use the serial link debug mode. In this debug mode the fuzzy calculation is done by the PC, so we can use all the features of the ST6 fuzzyTECH tool. This debug mode has two advantages: we can visualize the application in real time and modify or optimize the process in real time. From the point of view of the ST6 executable file the fuzzy module is replaced by a communication module. Every time we perform a loop data is sent to the PC and an answer is waited for from the PC.

To implement the serial link we have to change the scale of every linguistic variable. The new scale will be in the range [0,255].

To run the serial link with the STEP1 design you have to modify the COM.DEF routine like this:

```
COM_TYPE    5          ; 3 (with timer), 5 (without timer)-> ser_real
COM_L       drc,7,6  ; PORT, TxD, SYNCHRO / RxD
MODE        24,8,1   ; BAUD, DATA, STOP
APPLI       1,1        ; INPUT, OUTPUT
```

We have to choose a baud rate less than 2400 baud because values greater than this do not allow the ST6 fuzzyTECH tool to run well with the serial link: the baud rate is too fast to update the different opened windows.

Then we have to generate the include file from the fuzzy module by typing the Compile To ST6 option of the ST6 fuzzyTECH tool.

SGS-THOMSON
MICROELECTRONICS

You have to run the command file STEP1.BAT with the option ser_real to generate the executable file:

```
step1 ser_real
```

Once you have generated the executable file you can run the application program using either the debugger or by programming the EPROM of the real microcontroller.

To run the serial link you have to reset the ST6 board and then put the ST6 fuzzyTECH tool in the serial link debug mode. If this procedure is not respected, an error message will exit you from the serial link debug mode each time you run your process on the ST6 board.

The ST6 FuzzyTECH tool performs fuzzification, rules inference and defuzzification.

Every operation can be displayed graphically.

Each time a fuzzy loop is performed, we sample the current temperature; we adjust the scale to get the delta input and by comparing the observed input with the terms we obtain a degree of truth which determines how much the observed input is really the hypothesis of the rules. This computation is called the fuzzification.

Then we can now formalize the fuzzy reasoning by using the fuzzy rules to deduce the fuzzy output. This operation is called the fuzzy inference. In the STEP1 design the inference is simple because the degree of truth of each input term is applied directly to each output term.

SGS-THOMSON
MICROELECTRONICS
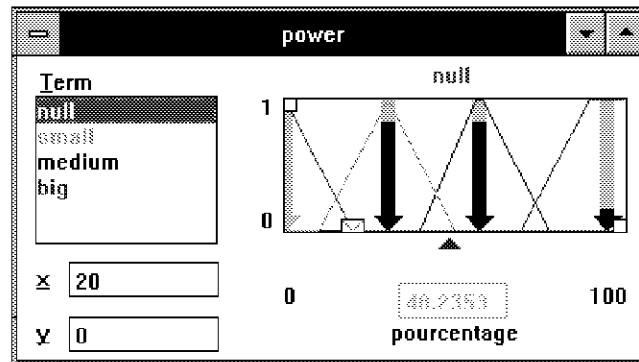
The last operation to perform is defuzzification. In this case we obtain a precise answer and consequently a precise action to be taken. The ST6 fuzzyTECH tool uses the Center Of Maximum Method (CoM) to compute the output. The CoM Method computes a crisp output as a weighed mean of the term membership maxima, weighted by the inference results.

Using the serial link debug mode, we have seen that the parameters of the system are convenient to regulate the temperature.

Once we have set the best compromise, we have to verify the fuzzy behaviour using the definitive PWM treatment. To do this we have to generate the ST6 fuzzy module using the Compile to ST6 option of ST6 fuzzyTECH. Then we link the communication module that allows to send data toward the PC with the application file. We can record data from the real process in a file on the PC and run it for analysis after conversion into the file recorder debug mode of ST6 fuzzyTECH.

Due to the fact that we have chosen the maximum counting time for the timer, communication time toward the PC does not matter because it will always be shorter than the timer count.

To do this respect the following instructions:

■ Modify the COM.DEF routine as follows:

```
COM_TYPE    4             ;  2  (with  timer),  4  (without  timer)  ->
ser_diff
COM_L       drc,7,4       ; PORT, TxD, SYNCHRO / RxD
MODE        96,8,1        ; BAUD, DATA, STOP
APPLI       1,1           ; INPUT, OUTPUT
```

■ Now run the command file STEP1.BAT with the option ser_diff to generate the executable file:

```
step1 ser_diff
```

■ Open a DOS window and change the parameters of the screen (to full screen and exclusive task) or exit WINDOWS; this is to have the maximum priority for the communication to avoid communication problems.

SGS-THOMSON
MICROELECTRONICS

■ Run:

```
FDBST6 -f temper1.ftl -run
```
on the PC (choose port number and 9600 baud)

■ Reset and run the application on the ST6 board

■ Type `START` (return) on the PC
 Data is recorded every time a loop occurs in a file called temper1.res

■ Now we can turn off the serial link debug mode constraint, this means that we can change the base variable of each linguistic variable to its original unit and save the application to the temp1.ftl file.

■ Generate a file called temp1.ptn that can be understood by the file recorder mode of the fuzzyTECH. To do this run:

```
FDBST6 -f temp1.ftl -ptn -i temper1.res
```

Using the .ptn file we can display the behaviour of the application. This is useful as it allows the display of the time response of the linguistic variables of the system.
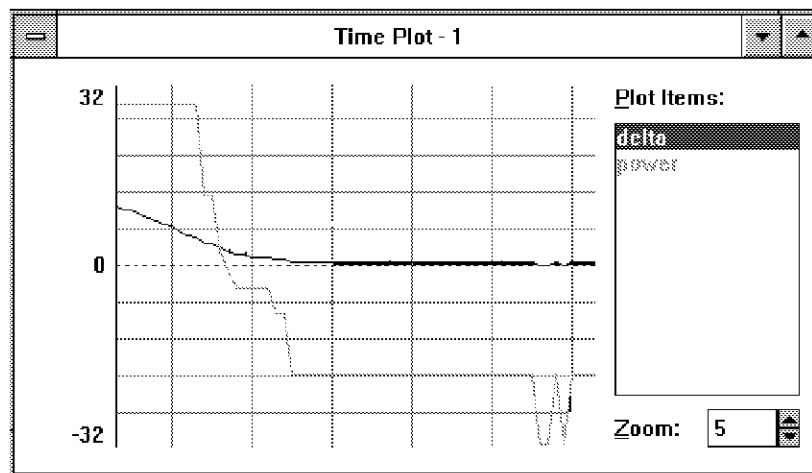
2.2 SECOND STEP

We can now introduce the second input that can be used when we are around the reference temperature; this variable is the speed at which the temperature reaches the reference DdTdt. Introducing this variable can give the system more accuracy and more stability.

Definition of the new variable of the system

This application aims at maintaining the current temperature to the reference. To do this we need information from the system: the difference between the current and the reference value and the speed at which the temperature reaches the reference. Each time we perform a loop (every time period) we calculate the variables of the system. The figure below shows an example of the regulation with all the parameters useful to build the system variables.

Figure 4 : Closed loop regulation
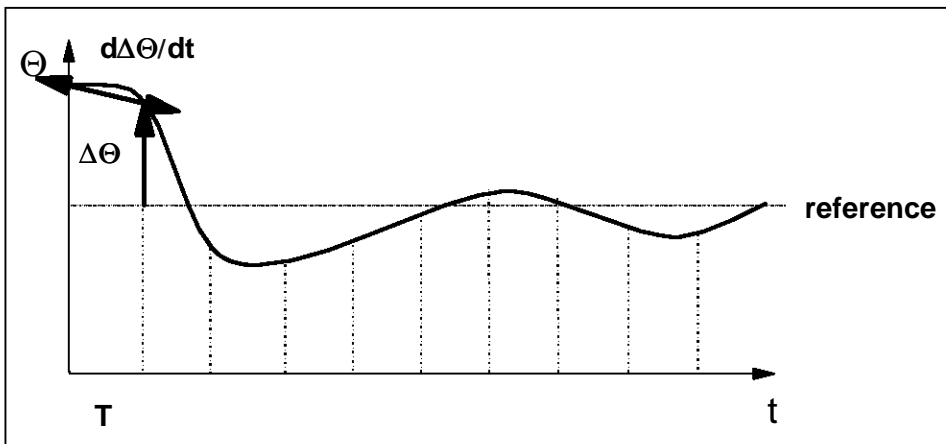
In this application we consider delta (DQ) as the difference between the current temperature and the reference and DdTdt (dDQ/dt) as the difference between the previous delta and the current delta.

As we did in the STEP1 design we define a set of terms for the linguistic variable DdTdt. The set of terms is [NEGBIG, NEGSMALL, NULL, POSSMALL, POSBIG]. The typical values, and maximum and minimum values are represented in the array below:

DdTdt = speed at which the temperature reaches the reference

Value	minimum value	typical value	maximum value
negbig	-4° // 0	-2 °// 64	-1° // 96
negsmall	-2° // 64	-1°// 96	0° // 127
null	-0.5 °// 112	0 °// 127	0.5 °// 144
possmall	0° // 127	1° // 160	2° // 192
posbig	1 °// 160	2 °// 192	4° // 255

The shell unit is degrees; we just measure the speed at which the temperature reaches the reference temperature. Values take part of the interval [-4y, +4y] which corresponds to [0,255] in code values. In the array each item is composed of two values: the shell value and the code value.

As we deal with small variations in temperature and as we want to use the entire range of the microcontroller resolution, we multiple the result by 8. Another reason is also that we use the entire scale for the definition of the terms of membership function.

Algorithm of the system

Using the speed to which the temperature reaches the reference we have built the following rules:

Figure 5 : Algorithm reference

To decrease the effect of the second variable we have chosen the MIN operator.

SGS-THOMSON
MICROELECTRONICS

Prototyping optimization

Using this second variable we have first experimented with the fuzzy model with serial debug mode, then, using the file recorder mode in the same conditions as before we have obtained the following time response for the system:

Comparing the two solutions (STEP1 and STEP2) we notice that by using STEP2 we reach the reference temperature at a quicker rate than with STEP1. But sometimes we have great variation of PWM values; so it is necessary to optimize the rules of the system to have small variation of the PWM power.

2.3 THIRD STEP

To fine tune the system and to get small variations of PWM, we add an extra term to the power output variable and we add another term to the delta input variable. We create the same set of rules as for STEP1, then, when we are next to the reference, we take into account the speed to which the current temperature reaches the reference (dDTdt).

Experimenting the system we notice that the dDTdt variable does not fluctuate greatly and, due to the fact that we have a scale factor for this variable, the membership functions can take only certain weight values, so it is necessary to adjust them to average the delta variable.

Using the MIN aggregation operator, terms of dDTdt variable has an influence on the system when the weight of delta terms are superior to the weight of dDTdt terms.

Figure 6 : Optimizing the system
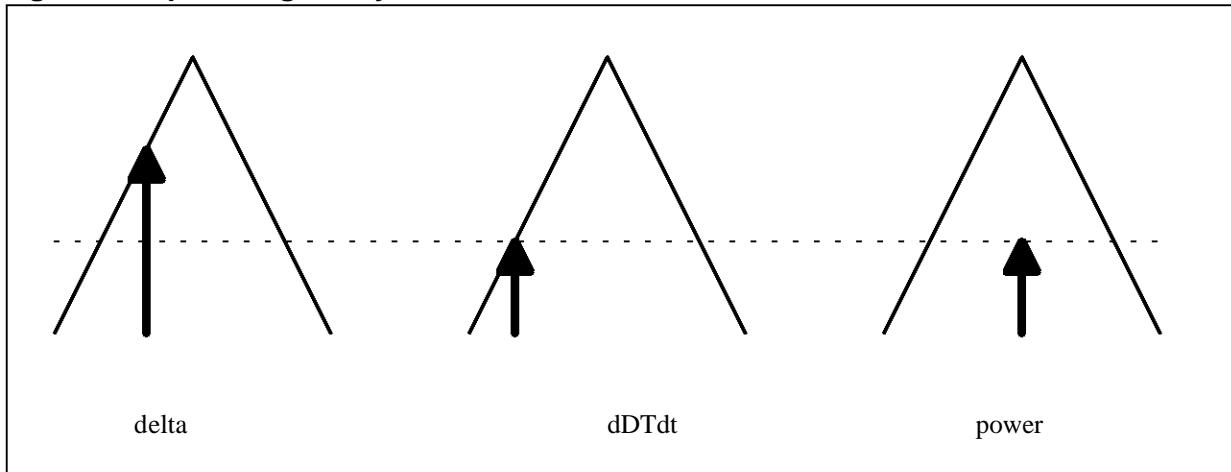
Using the file recorder mode we record data into a file and then using the spreadsheet rule editor and the timeplot we can see any discontinuities in the PWM action. The solution to avoid a discontinuity is to add another rule or to change a membership function.

Once we have changed the system, we make another executable file, record data into a file again, and then use the file recorder mode. This loop of modifications can last as long as necessary to reach the optimum variation of PWM.

The final set of rules used to describe the system are shown below:

Matrix	IF		THEN	
Utilities	**DdTdt**	**delta**	**DoS**	**power**
1		negbig	1.00	null
2		negsmall	1.00	null
3		posbig	1.00	verybig
4		possmall	1.00	medium
5		zero	1.00	null
6		posmedium	1.00	big
7	possmall	possmall	1.00	big
8	null	zero	1.00	medium
9	possmall	zero	1.00	medium
10	negsmall	zero	1.00	null
11	null	possmall	1.00	big
12		posmedium	1.00	verybig
13				

SGS-THOMSON
MICROELECTRONICS

The time response of the system for the final set of rules is described below:

3 SUMMARY

This note is a tutorial to get started with fuzzy logic. It presents a temperature control process that gives acceptable results for home appliances.

Fuzzy logic is used widely for embedded control because it provides simplicity, sensitivity, robustness, and easy optimization. Fuzzy logic is well known across the consumer market. It aims at increasing product performance and reducing development time. In this way, a product can be taken to the market more quickly, giving a competitive edge to the application.

SGS-THOMSON
MICROELECTRONICS