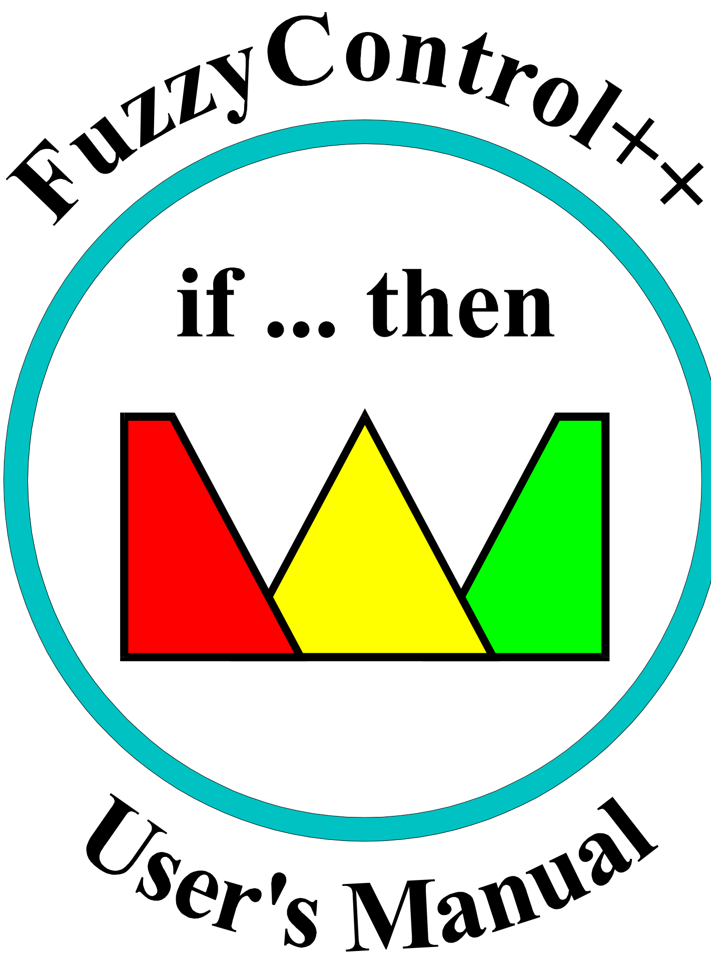
SIEMENS

FUZZY CONTROL ++V5

FuzzyControl++ User's Manual



FUZZYCONTROL++

Fuzzy configuration tool, consisting of:

- Configuration tool for PC or PG
- Runtime modules for various target platforms

Version 5.0

Siemens AG, 2003

- User's Manual -

Siemens AG, I&S IT PS 213

PO Box 32 40

D-91052 Erlangen

PART I:

INTRODUCTION

TO THE

SUBJECT MATTER

This part is subdivided into three chapters and its aim is to provide a brief and easy to understand introduction to fuzzy logic, neural networks, and the FUZZYCONTROL++ program.

For clarity's sake, theory is kept to a minimum and is illustrated with simple, practical examples and diagrams.

1 Fuzzy Logic

1.1 History of Fuzzy Logic

Despite its name, fuzzy logic denotes a precise idea, as we shall see below. The notion of "fuzziness" in logic was pioneered by J. Lukasiewicz [17] and M. Black [2] as early as 1935.

The idea came about because it was recognized that in many cases a binary assignment true/false, yes/no, on/off is inadequate to describe real world situations. (Consider the question: "Is a car which travels at 60 mph fast or slow?" - yes or no?). In 1965, L. A. Zadeh [27] extended classic binary set theory (yes/no) with his "fuzzy set theory". He was the founder of "fuzzy logic", which can be seen as a sort of generalization of the binary logic known as Boolean algebra. On this basis, Mamdani in 1973 used fuzzy logic to control processes for the first time, which led to the development of fuzzy control as a special branch of fuzzy theory.

The importance of Zadeh's thinking initially went unrecognized. It was only many years later - in 1988 - that Japanese scientists and engineers first put fuzzy set theory into practical use [29]. After that, the fuzzy tool finally aroused worldwide interest. From 1991 onwards, fuzzy logic was used for various applications in the USA and Europe, too, and is now an established problem-solving method.

1.2 Fundamentals of Fuzzy Logic

Non-fuzzy and fuzzy sets

Classic set theory makes every element *a member of* or *not a member of* a set. Membership is therefore limited to just two values (yes/no, 1/0, true/false). Sets with such binary membership functions are called non-fuzzy sets. Fuzzy logic, on the other hand, expands the membership function to allow intermediate values and thus better simulates human thinking.

Human beings use concepts , which have a gradual transition to the opposite meaning. For example, the terms "warm" and "cold" are temperature ranges with fuzzy boundaries, and they can even overlap. "Warm" and "cold" are therefore fuzzy sets.

As an example, lets consider the values of the temperature scale in °C as elements to be assigned to defined sets: One possible set of temperature values is called "frost". Frost denotes a non-fuzzy set of temperatures, i.e. all temperatures below 0 °C. Membership of the temperature "-5 °C" in this set is therefore "one", or "yes", and membership of temperature "+5 °C" is "zero", or "no".

The notion of "cold" denotes a fuzzy set because, while temperatures around 0 °C may be perceived as being 100% "cold", +5 °C is less "cold" and +10 °C is not "cold" at all, indeed it could be a member of a new fuzzy set called "lukewarm" ranging from +5 °C to +15 °C.

Applications of fuzzy logic

Information processing based on fuzzy logic is appropriate, often necessary, where processes are specified by verbally expressed algorithms in the form of "IF-THEN rules". For example, in complicated manufacturing processes it can be observed that operators take control actions on the basis of their experience using simple if-then criteria ("If the temperature is high and the pressure is medium, then I must open the valve halfway.") and using this way, they often achieve good operation with quality results.

The applications of fuzzy logic are not limited to closed-loop and open-loop process control; fuzzy logic is also used for information processing in measured value acquisition (pattern recognition, signal processing) and in operations management and planning (scheduling, forecasting). Fuzzy logic makes it possible to transfer the "experience" of an expert to a computer and obtain an initial and "automated" - though perhaps not yet optimal - solution quickly.

Advantages and disadvantages of fuzzy logic

The fundamental advantage of fuzzy logic is that no mathematically expressed description of the process to be automated, in the form of algebraic equations or differential equations etc, is required to design and use fuzzy systems.

However, not using a mathematical model of the process is also a source of uncertainty. For example, the many degrees of freedom in designing fuzzy systems can be a special disadvantage.

The following overview provides a comparison:

☺ Simple implementation of verbally expressed rules (if ..., then...) on a computer to solve a problem.

☺ The behavior of the fuzzy system is understandable to human beings.

☺ Avoids the costly development of a mathematical description when compared with conventional methods.

☺ Possible to use for processing complicated and involved processes.

☹ Task definitions with not enough knowledge of the system and little or very imprecise knowledge of the system behavior result in bad, possibly unusable, fuzzy solutions.

☹ Usually no adaptability and learning capability if the system behavior changes.

☹ Design of a system requires experience because of the many degrees of freedom.

1.3 Fuzzy Control

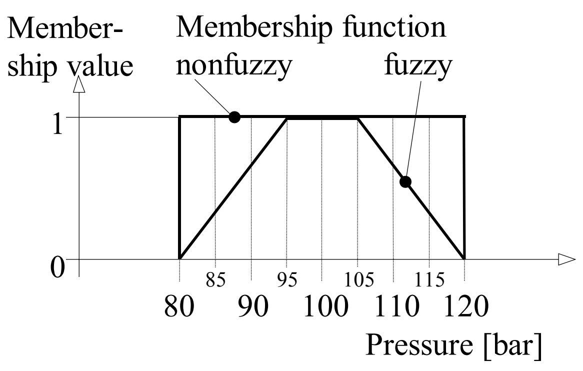
Through experience of using fuzzy logic for information processing in open- and closed-loop control (fuzzy control), especially in "fuzzy rule bases", tried and tested methods have emerged. The most important of these are explained below.

The membership function

An introductory example: Pressure monitoring in a processing plant

The operator of a processing plant determines a pressure of 100 bar as being optimal for the production process. However, he allows an operating range of 80 to 120 bar as acceptable ("Pressure OK"). The following diagram shows the difference between the set of pressure values "Pressure OK" defined both as a non-fuzzy set and a fuzzy set. The representation of the membership values (degrees of membership) via the elements of the set (here pressure values) is the *membership function*.

In general, membership can be defined by the most varied functions (such as normal distribution, sigmoid function, trapezoidal function). For practical purposes, *triangular* and *trapezoidal functions* have become the standard. They can be described with 3 or 4 points and therefore do not cause much computational overhead.



If you imagine how the operator works, you can see the advantage of modeling with fuzzy logic. The operator prepares to take action when the pressure display reaches 115 bar because the pressure can only be considered 30% OK. A binary control with a threshold value contact at 120 bar would not be able to perform such qualified operation and would perceive the situation all of a sudden when the pressure value left the non-fuzzy set. The operator, on the other hand, can recognize critical values and tendencies early on and can therefore react in time taking any necessary counter-measures. To imitate this behavior it is necessary to use a fuzzy system instead of a binary control.

Function of a fuzzy system

A fuzzy system processes its information fuzzily in the form of rules, such as

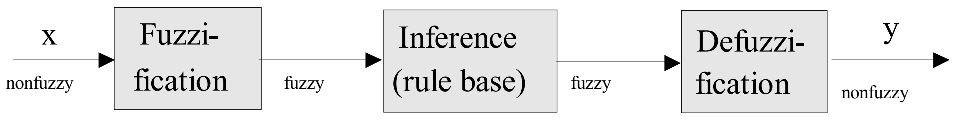
IF input value "pressure OK", THEN output value "do nothing", or

IF input value "medium", THEN make output value "small".

The input and output values are *linguistic variables* and the terms "medium" and "small" are not numeric but *linguistic values*. They are also called *fuzzy sets.* A *linguistic variable* usually possesses 3, 5 or 7 *linguistic values*. The complete collection of rules forms the *rule base*.

However, the fuzzy system is, necessarily, part of a technical system that works with numeric values (signals), i.e. with non-fuzzy values, at its inputs and outputs. So appropriate conversion must be performed at the input and output of the fuzzy system. This conversion is termed fuzzification or defuzzification.

The following diagram illustrates the principle.



- **Fuzzification**

  The degrees of fulfillment for the linguistic values (degrees of membership of the fuzzy sets) of the linguistic variables are assigned to the non-fuzzy input values. So fuzzification determines, for example, to what degree the "pressure is OK" if it is, say, 115 bar. This is done using the membership function. This degree is called "Degree of fulfillment of the IF part".

- **Inference**

  For each rule of the rule base, the degree of fulfillment of the THEN part is formed from the degree of fulfillment of the IF part by a certain method. This process is also called *implication*. The degree of fulfillment of the THEN part is equivalent to the degree of fulfillment of the rule, which is also called the rule intensity. All these individual rule evaluations put together result in one membership function for the output signal, which is also termed *composition*. The resulting membership function describes a "fuzzy control command" that could, for example, be something like "Position valve just below the center".

  When the IF part contains a combined statement "IF.... AND..." --- (THEN....), the fuzzy logic AND operation is executed first and the degree of fulfillment is used in the overall rule evaluation. All these statements together are often called an *aggregation*.

- **Defuzzification**

  The most representative numeric (non-fuzzy) output value is calculated for the control variable from the fuzzy control command (in the form of the resulting membership function).

The following example illustrates these three typical function blocks. Here more emphasis is placed on clarity than creating realistic values.

Example: Acceleration control on a vehicle:

- Aim: Automatic acceleration and braking adapted to the situation of the vehicle
  The following definitions are required before we can *design the fuzzy rule base*:
- Input signals of the rule base:
  1. Distance from the vehicle in front (linguistic variable "Distance")
     linguistic values: "small", "medium", "large"

2. Own current speed (linguistic variable "Speed")
   linguistic values: "slow", "medium", "fast"

- Control Variables:
  Acceleration a (positive - accelerate or negative - brake) of the vehicle (linguistic variable "Acceleration")
  linguistic values: "neg_large", "neg_small", "zero", "pos_small", "pos_large"
- The rule base:

R1:	IF Distance = small	AND Speed = fast,	THEN a = neg_large;
R2:	IF Distance = small	AND Speed = medium,	THEN a = neg_small;
R3:	IF Distance = medium	AND Speed = fast,	THEN a = neg_small;
R4:	IF Distance = medium	AND Speed = medium,	THEN a = zero;
R5:	IF Distance = medium	AND Speed = slow,	THEN a = pos_small;
R6:	IF Distance = large	AND Speed = medium,	THEN a = pos_small;
R7:	IF Distance = large	AND Speed = slow,	THEN a = pos_large;

  The rules are also often represented in a matrix.

- Definition of the membership functions (fuzzy sets) for the linguistic values:

  In this design step, the form of the fuzzy sets is selected along with their *range of influence*, i.e. the range of non-fuzzy values of the input and output signal is determined for which the fuzzy set has a degree of membership greater than zero.

- Definition of the inference and defuzzification procedure:

A number of methods exist. The common methods include MAX-MIN inference and MAX-PROD inference and defuzzification according to the centroid method. In practice, the choice depends on the design tools the computer program provides. In *FUZZYCONTROL++* a special method is used that only demands a low computational effort in the practical implementation of the fuzzy system. The principle is explained in the following section "how the fuzzy rule base works".

To explain *how the fuzzy rule base works,* let us look at an "operating case" in which the speed is 65mph and the distance is 51m. The membership functions are shown in the following two diagrams. In choosing the curves of the membership functions, the available degrees of freedom must be defined in a meaningful way (in the form of "intermediate points" of the triangles or trapezoids). For example, the fuzzy set "medium" of the input signal Distance refers to the range 30 to 120 m.

- Fuzzification:

The following can be seen from the curves of the membership functions chosen here for the two input signals:

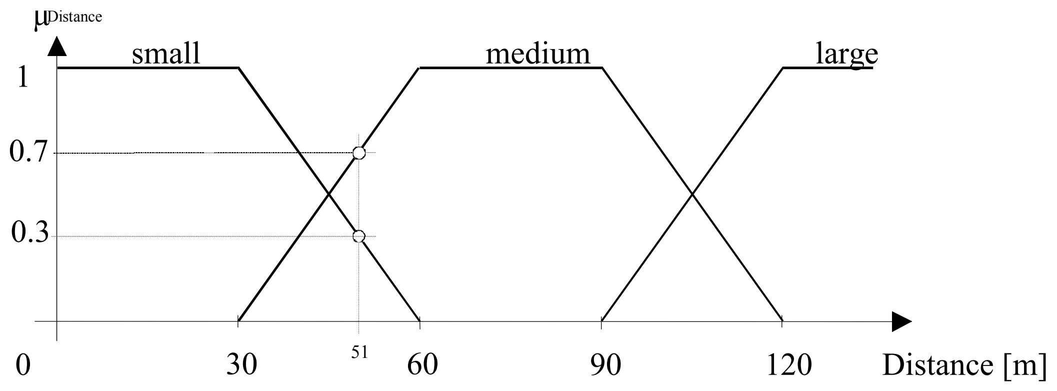
The fuzzy statements about the distance, which is 51 m, are 30% true for "small" and 70% true for "medium". (The statement "large" is 0% true. Such cases are usually not pursued.) Similarly, the degrees of membership for "medium" (0.8) and for "fast" (0.2) result for the fuzzy sets of the "Speed" variable.

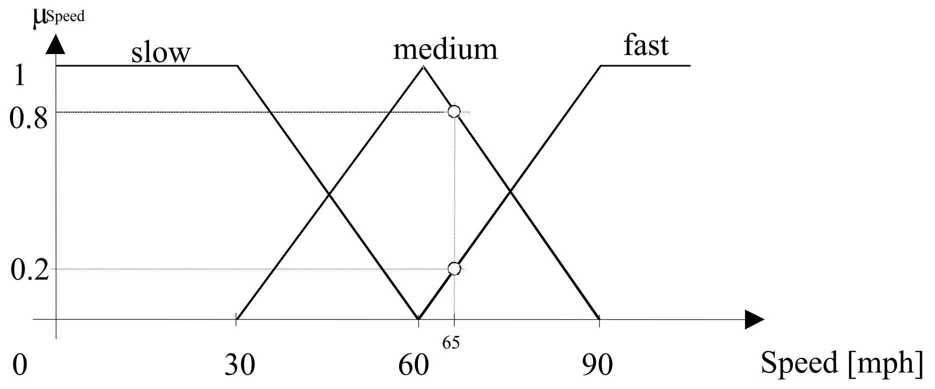
- Inference:

Let us consider rule 1 as an example:

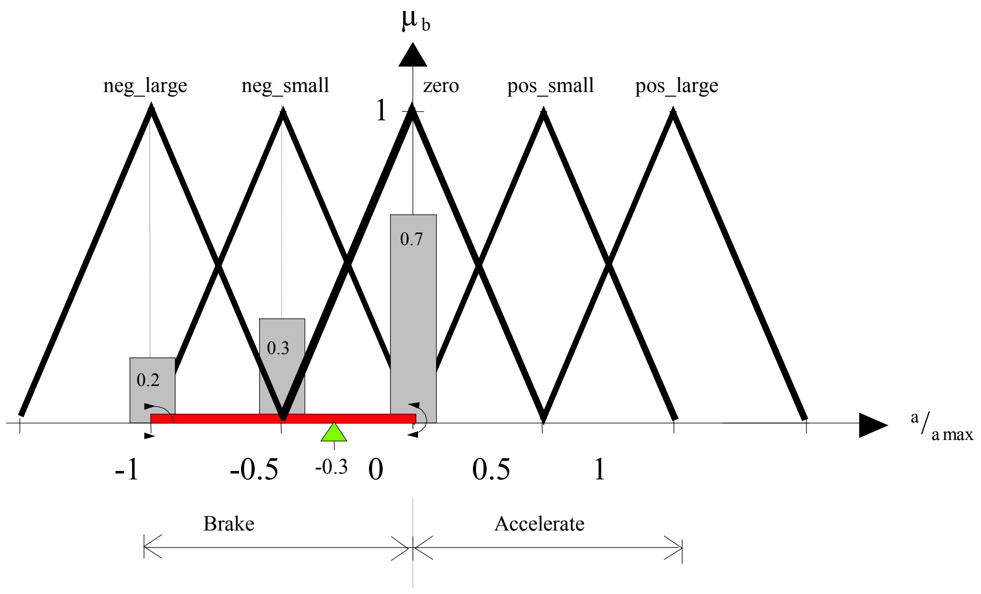
In the IF part of the rule, the fuzzy statements about the two variables Distance and Speed are ANDed. Distance is "30% small" and Speed is "20% fast". The entire IF part of this rule is therefore considered to be 20% fulfilled after application of the MINIMUM operation for fuzzy logic AND combination (aggregation).

The THEN part of rule 1 refers to the fuzzy set "neg_large" of the output signal. The simplest implication method makes the (sensible) assumption that an implication can be fulfilled to no greater degree than the antecedent. Because the IF part is 20% fulfilled, it can be inferred that an acceleration is required which is 20% "neg_large" (implication). This degree of fulfillment must now be used to modify the membership functions of the output signal in the THEN part of rule 1.





The following diagram shows the output signal "Acceleration" with its 5 fuzzy sets. The fuzzy set "neg_large" is modified by the 20% degree of fulfillment from rule 1. One simple method is to consider only the height of the peak. This makes the membership functions vertical lines with height 1, i.e. 100% at the position on the output signal axis where the peak is located. (This type of membership function is called a *singleton*). All that rule evaluation does is to shorten the line, in this example, to 20%, as shown in the diagram.



If you add the remaining rules to the evaluation for the same pair of non-fuzzy input values (51m and 65mph), the following degrees of fulfillment result:

Rule 1:	The command: Make acceleration "neg_large" is 0.2 fulfilled
Rule 2:	The command: Make acceleration "neg_small" is 0.3 fulfilled
Rule 3:	The command: Make acceleration "neg_small" is 0.2 fulfilled
Rule 4:	The command: Make acceleration "zero" is 0.7 fulfilled

In this case, rules 5 to 7 are not fulfilled, because at least one IF condition is not fulfilled, i.e. the degree of membership is zero for the current non-fuzzy input value (e.g. rule 5 with Speed slow).

The control recommendation of the fuzzy rule base consists of the above four partial statements in this operating case. (Please note that two different recommendations for the same fuzzy set "neg_small" result from rules 2 and 3.) Composition into a resulting membership function is best performed by superimposing all these "partial membership functions" (composition). One possible operation is the MAXIMUM operation. In this way, the result of rule 2 covers the result of rule 3 and the resulting control recommendation consists of the three columns shown in the diagram. With *FUZZYCONTROL++*, the SUM operation is used so that a rule intensity of 0.5 results from rules 2 and 3 at position $a/a_{max} = -0.5$.

However, the control recommendation resulting from all rule activity is still fuzzy and might be expressed as "on no account full braking but not quite half the braking power".

- Defuzzification:

From this control recommendation, which is still fuzzy, we now have to calculate a non-fuzzy, numeric value for the control variable as a numeric value (from the manipulating range). It must be representative of the fuzzy control recommendation, i.e. the resulting membership function. For example, if the resulting membership function consists of the superimposition of partial areas of triangles and trapezoids, the position of the center of gravity of the total area on the output signal axis is the non-fuzzy value sought. In this case (see diagram), you can imagine a mechanical system in which the three columns for acceleration values, whose weight is a function of their height, are balanced on a pivot along the axis ("equilibrium"). This is the principle on which the centroid method is based.

In this case, it is possible to find the non-fuzzy value, i.e. the numeric control variable by the following simple calculation:

$$\frac{\sum a_v \cdot \mu_{a_v}}{\sum \mu_{a_v}} = \frac{-1 \cdot 0.2 - 0.5 \cdot 0.3 - 0 \cdot 0.7}{0.2 + 0.3 + 0.7} \approx -0.3$$

In this example, the vehicle’s brake is applied with approx. 30% of the maximum possible range.

2 Neural Networks

2.1 History of Neural Networks

The first neural networks were developed by biologists in the hope of being able to simulate natural neural networks. The term "neuron" is used in medicine and biology to denote the nerve cell. Mathematicians and engineers have adopted this model for use in mechanical information processing. These "artificial neural networks" are a greatly simplified attempt to imitate the function of nerve cells as found in lower-order organisms. When we talk about a "neural network" we mean an "artificial neural network". Like biological brain structures neural networks have the ability to learn.

The origins of artificial neural networks go right back to the early 1940's.
In 1943, W. S. McCulloch and W. Pitts developed a model ("MP neuron") that imitated the function of human nerve cells [5]. This greatly simplified abstract neuron model already shares the following important functional properties with a nerve cell:

• Structure:	Many inputs (synapses) and one output (axon).
• States:	Two possible states (inactive or activated).
• Connection:	The neurons are interconnected (networked).
• Independence:	The state of a neuron only depends on its input activation.
• Activation condition:	A neuron is stimulated if enough inputs are activated.

D. O. Hebb formulated the first learning method in 1949 [9] ("Hebbian learning rule"). Neural networks capable of learning were created on this basis and finally in about 1955, the first computer simulations were implemented. P. J. Werbos [25] and D. E. Rumelhart [18] developed the backpropagation learning algorithm , which is the most frequently used learning method.

On the basis of this work the theory of artificial neural networks was elaborated still further. After 1970, neural networks with the associative memory function were created and T. Kohonen developed self-organizing feature maps [16]. After 1985 the further development and application of neural networks really took off.

Another - non-biologically inspired - approach was networks based on the radial basis functions (RBF). They originated from mathematical approximation theory and were first used in 1988 by D. S. Broomhead and D. Lowe [4].

2.2 Fundamentals of Neural Networks

The usefulness of neural networks stems from their ability to learn, i.e. their ability to simulate the behavior of a process from a collection of input and output data about a process (procedure, functionality). The range of applications, technical and non-technical, to which they can be applied is therefore very wide.

2.2.1 Applications and Properties

Neural networks can be used for problems, whose structure and solution are not known or are only partially known. Using an example, the neural network learns the solution to a problem. This type of information processing is therefore fundamentally different from a conventional programmed system.

Applications of neural networks

Neural networks are typically used for the following tasks:

- Pattern recognition (e.g. for machine reading of handwriting)
- Identification of characteristics or processes
- Filtering of data (e.g. for intercontinental telephone systems)
- Data evaluation (e.g. for diagnostic systems)
- Data interpolation
- Closed-loop and open-loop control of processes

Advantages and disadvantages of neural networks

☺ The great advantage of neural networks is solving a problem using example data. This data might, for example, be available in the form of measurement series.

☺ Another strength of neural networks is their adaptability, i.e. they can adapt to a new situation by changing their behavior.

☺ A neural network is especially suitable for simulating complex and nonlinear dependencies.

☹ One disadvantage of neural networks is that it is generally not possible to understand how they have solved a problem.

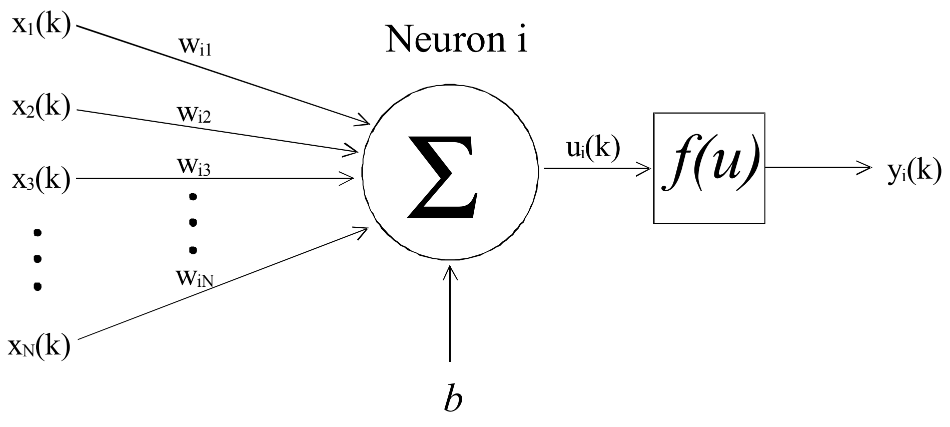
☹ The network is no "smarter" than the data you "trained" it with, i.e. the example must be an adequate representation of the problem.

2.2.2 The nature of neural networks

Relatively simply structured elements (neurons) receive data from numerous neighboring elements with which they are linked via weighted connections and associate this data according to simple rules. Although the complexity of each element is relatively low, interlinking them can considerably increase the power of the network as a whole.

The neurons are the individual elements of the neural network The ordered links between them constitute the structure of the network. The fundamental principle of the neural network can be illustrated using the example of an individual neuron (see the diagram below).

The neuron



The above diagram is a generalization of the MP neuron developed by McCulloch & Pitts as early as 1943.

A neuron *i* has several inputs $x_j(k)$ (e.g. j = 1 ... N) and 1 output $y_i(k)$, where k is the time at which the function of the neuron is considered. Information is available at the inputs and the output in the form of (initially) random real numbers. The sum of the numbers at the input, which might be derived from measurements, is also called the input pattern or input vector. The connection lines show that the neuron exhibits an input/output characteristic, in which the output value only depends on the input values and not vice versa.

(The output can also be "fed back" to produce an "additional input". This "neuron with internal feedback" is not discussed further here.)

The behavior of neurons is characterized by two properties: First the weighted sum $u_i(k)$ of all inputs is calculated from the weights (factors) on the connection lines (edges) w_{ij} . The value at the output is then derived from this sum using the activation function f(u) . The variable b (bias) only causes a shift in the y-axis of the activation function. The activation function and the weights have a major influence on the behavior of the neurons.

A neuron is considered *active*, if its output has a certain value, e.g. exceeds a threshold value. This activity is the result of *stimulation* of the neuron by its inputs and the weights of the connections. The weights can promote stimulation if they are greater than zero - or inhibit it if they are less than zero. If the weight is zero, the input in question has no affect on the activity of the neuron.

It is possible to express the behavior of the neuron mathematically:

For the weighted sum, the following applies initially:

$$u_i(k) = \sum_{j=1}^{N} x_j(k) \cdot w_{ij}(k) + b\,;$$

The overall input/output characteristic y = g(**x**) of the neuron *i* can be represented by the following formula:

$$y_i(k) = f\left(\sum_{j=1}^{N} x_j(k) \cdot w_{ij}(k) + b_i\right)$$

The most common *activation functions* are:

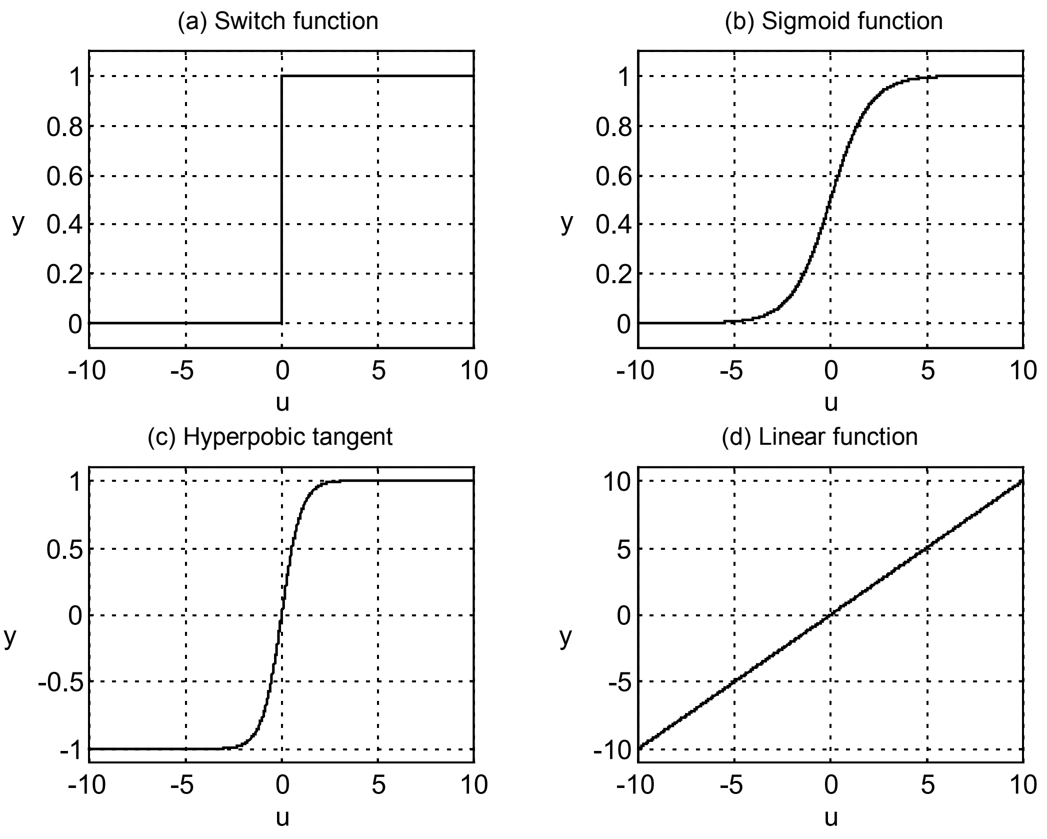
(a) Switch function: $y = \begin{cases} 1, & \text{for } u > 0 \\ 0, & \text{else} \end{cases}$

(b) Sigmoid function: $y = \dfrac{1}{1 + e^{-u}}$

(c) Hyperbolic tangent: $y = \tanh(u) = \dfrac{e^{u} - e^{-u}}{e^{u} + e^{-u}}$

(d) Linear function: $y = u$

The following diagrams show the curve for each function.



For the function of the neuron it is therefore characteristic that both the input values and the weights have an influence on its output value. Moreover, for most applications, activation is nonlinearly dependent on the weighted input sum and can be either sudden (Fig. a) or gradual (Figs. b, c, d).

A neuron therefore has the ability to assign an output value to a certain input pattern that (for this pattern and a given activation function) depends only on the current weighting.

Networking of many neurons

If several neurons are interconnected, the structure formed is called a neural network. Each neuron contributes to the input/output characteristic of the overall neural network. We can therefore talk of the "knowledge" of a neural network being distributed throughout the entire network structure. It is the job of the learning method used to set and optimize the neuron weights for the problem in hand.

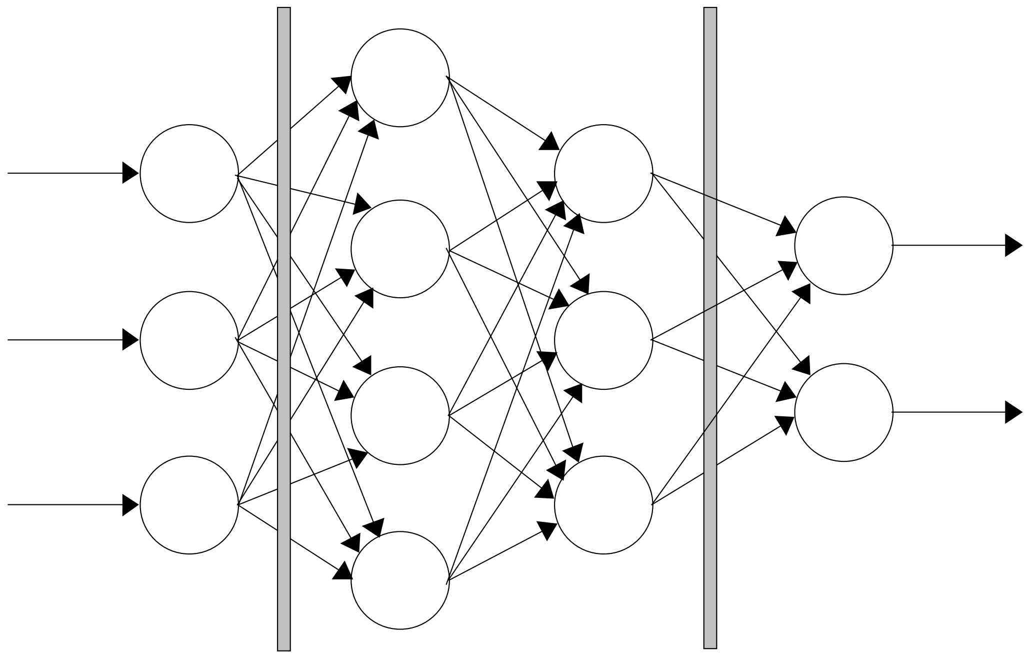
In principle, any links between neurons are possible and this has a strong influence on the behavior and properties of the network. In a neural network with only feedforward links, the input/output characteristic has the property that the output pattern at a specific point in time only depends on the input values and weights at that point in time. The time period that is required to transport and process the information in the network can usually be neglected. Networks of this type are called *static networks*. Static networks and their properties have now been well researched and are already being used for a wide range of applications.

Dynamic networks usually arise because internal feedback has been introduced so that the output of the neuron at time k depends on its own output at time k-1 (the output is an additional input). Dynamic networks are still at the research stage and are rarely used in practice because of the great difficulty in verifying their stability and the complex learning methods involved. However, there is another way of taking dynamic processes, such as changes in the input signals or the use of past values, into account: The dynamic input signals in question are simply made available to the static network as additional inputs (e.g. change in a value between time k-1 and time k).

2.2.3 The multilayer perceptron (MLP network)

Probably the best-known type of neural network is the *multilayer perceptron* (abbreviated to MLP). In this case, several neurons are interconnected by feedforward links without any internal feedback of their output signal. This is called a feedforward network and is an expansion of the classic perceptron (which only consists of a single neuron). The MLP is a static network.

In the MLP network, the neurons are arranged in several layers. The two layers that connect the structure with the outside world are called the input and output layers. The layers in between are called *hidden layers*). The following diagram illustrates this structure.



Input layer Hidden layers Output layer

The neurons of the input layer are only used to distribute the inputs to the neurons of the first hidden layer. Each therefore only has 1 input that is assigned to the input value of the network at this position. The weight of the input connection can be used to scale the input signal. In this respect, input neurons are a special case of the neuron. Some authors do not therefore include the input layer in the number of layers of a network. In *NEUROSYSTEMS,* however, the input layer is counted.

The set of all input values at a particular point in time is the "input pattern" (input vector) of the network. The output values of all neurons of the output layer at the same point in time are the associated "output pattern" (output vector) of the network. The output pattern with which a network reacts to a specific input pattern is defined as the input/output characteristic of the network. It is also called the "response characteristic".

The connections between the neurons within the network are characterized by the fact that the output of each neuron of one layer usually branches to all the neurons of the next layer. Each output of one layer is therefore an input of all the neurons of the next layer. A following neuron therefore has as many inputs as it has preceding neurons. If a link has weight zero, it can be considered "non-existent" (in this particular case).

To summarize the above we can say:

A neural network consists of a number of interconnected neurons arranged in layers, resulting in a certain *architecture.* The *parameters* of the neural network are the characteristic values of the activation functions and the weights of the connections. The neural network can learn its *response characteristic*.

2.2.4 Radial basis function network (RBF network)

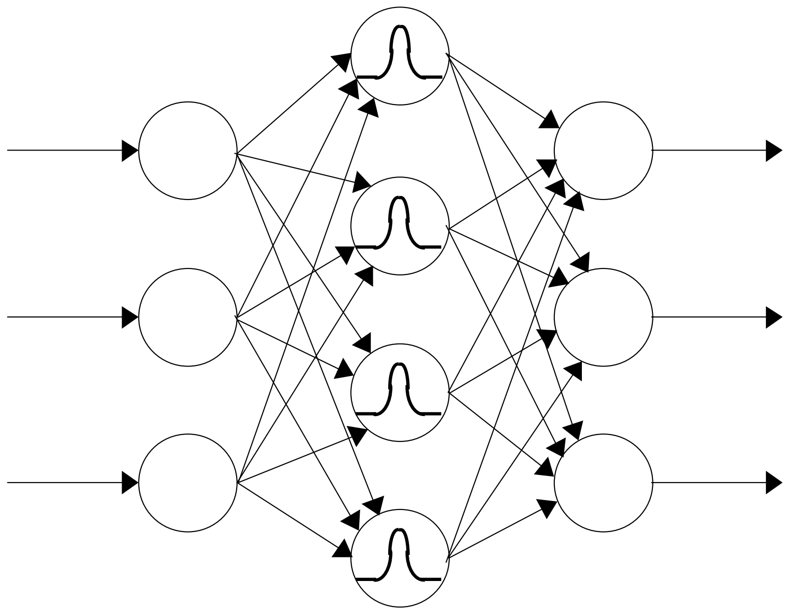
The method used in RBF networks is a little different from that used for the multilayer perceptron.

They always consist of three layers:

- The input layer,
- The second layer with RBF neurons whose activation function is the Gaussian function (as shown in the diagram below),
- And the output layer.

This is only a short outline of the principle. For more details, please consult the references [8].

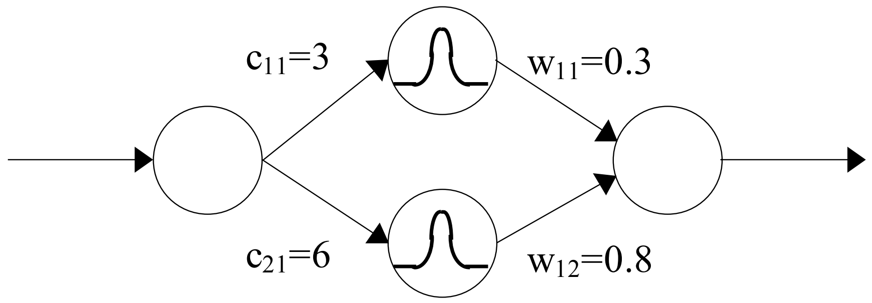
Here too, the connections between the input and the RBF layer are provided with weights c_{ij}. Unlike the MLP network, the weights are not multiplied with the inputs (that is the outputs of the neurons of the input layer). Instead, the RBF neurons process a measure of the similarity between the weight and the input value. The result is that each neuron of this second layer is stimulated more strongly, the more similar the input values and the corresponding weights are.



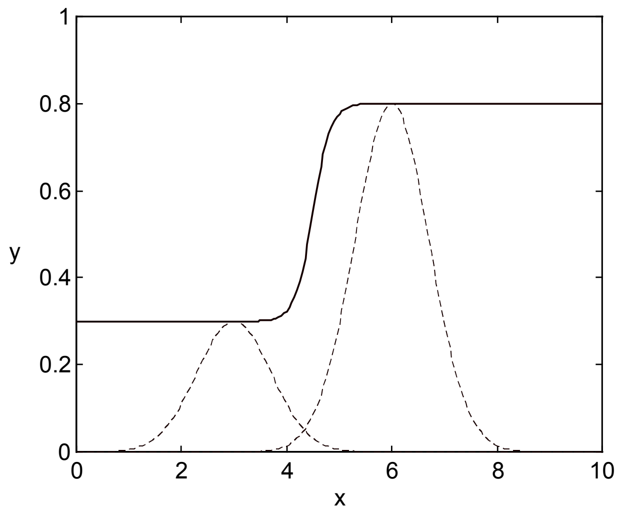
At the connections between the RBF layer and the output layer there are weights w_{ki} that are multiplied with the outputs of the RBF neurons, which results in additional weighting ("The neuron whose weights are most similar to the input values determines the output").

Each RBF neuron can therefore be seen as a sort of rule: If the inputs are similar to the weights cc_{ij}, the output of the network is similar to weight w_{ki} between the i^th RBF neuron and the k^th output.

This can be illustrated with a simple example with one input (j=1), two RBF neurons (i=1, 2) and one output (k=1):



Two RBF neurons are placed at positions x=3 and x=6. The inference (weights) of the two neurons are y=0.3 and y=0.8. This results in the following output curve:



In this type of network, the position of the bell-shaped functions (weights c), their width and their inference (weights w) are learnt.

2.3 How Neural Networks Work

A neural network characterized by two stages: *learning* and *reproduction*. The latter is the actual working phase in which input data is processed to form the required output data (e.g. text recognition). The network can only do that if it has first learnt how. This is called "training" and is done with "suitable training data". Training data or learning data are the names given to the input pattern and corresponding output patterns that represent the input/output characteristic required. Whether this data is "suitable" or not depends on the type of network and on the required input/output characteristic of the network.

So, how does a neural network learn?

You will remember from the previous chapter that the input/output characteristic of a neuron can be changed by the connection weights and the parameters of the activation function.

This gives rise to different methods:

- Learning methods that only correct the connection weights of the neurons without changing the actual structure of the neural network (e.g. the backpropagation algorithm)
- Learning methods that not only change the weighting of connections but also change the structure of the neural network, i.e. add or remove neurons (e.g. cluster methods).

Most learning methods just alter the weighting depending on success or failure during training. These learning methods are, in principle, nothing other than a - usually - nonlinear optimization problem in which the network weights are the parameters to be optimized.

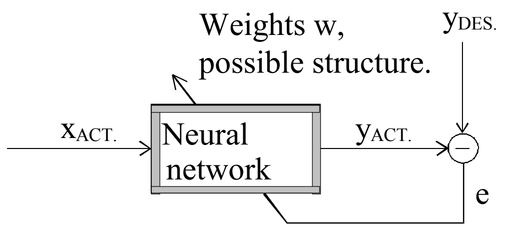
One of the possible training methods is "supervised learning". Supervised learning data can only be used with sets of learning that consist of a number of input patterns with corresponding output patterns (desired output patterns). Supervised learning is performed as follows:

- The network calculates the output values from the input values provided using the current input/output characteristic. The result is a current actual output pattern.

- The deviation between the actual pattern and the desired pattern is used to update the input/output characteristic of the network in such a way that the error is smaller after the next calculation of the output pattern.

- These two processes form one learning step. The learning steps are repeated cyclically until an error threshold, or a set number of steps or a set time is reached.

The following diagram illustrates the learning process.



The question remains as to how the neural network is corrected so that it really "learns", i.e. how the deviation between the desired and the actual value at the output of the neural network is minimized. In the simplest case (the backpropagation method), the contribution of each weight to the desired/actual value deviation is calculated and in a second stage, the weights are changed such that the error becomes smaller. The name of this method comes from the fact that the error component is "propagated back" to the weights causing the error.

There are numerous learning methods which we shall not explain in detail here. For more details, please consult the references, e.g. [7], [28].

3 Neurofuzzy Systems

3.1 Principle

Chapters 1.1 "History of fuzzy logic" and 1.2 "Fundamentals of fuzzy logic" have shown that the problems for which neural networks and fuzzy logic are used are fundamentally different. With some simplification, we can say:

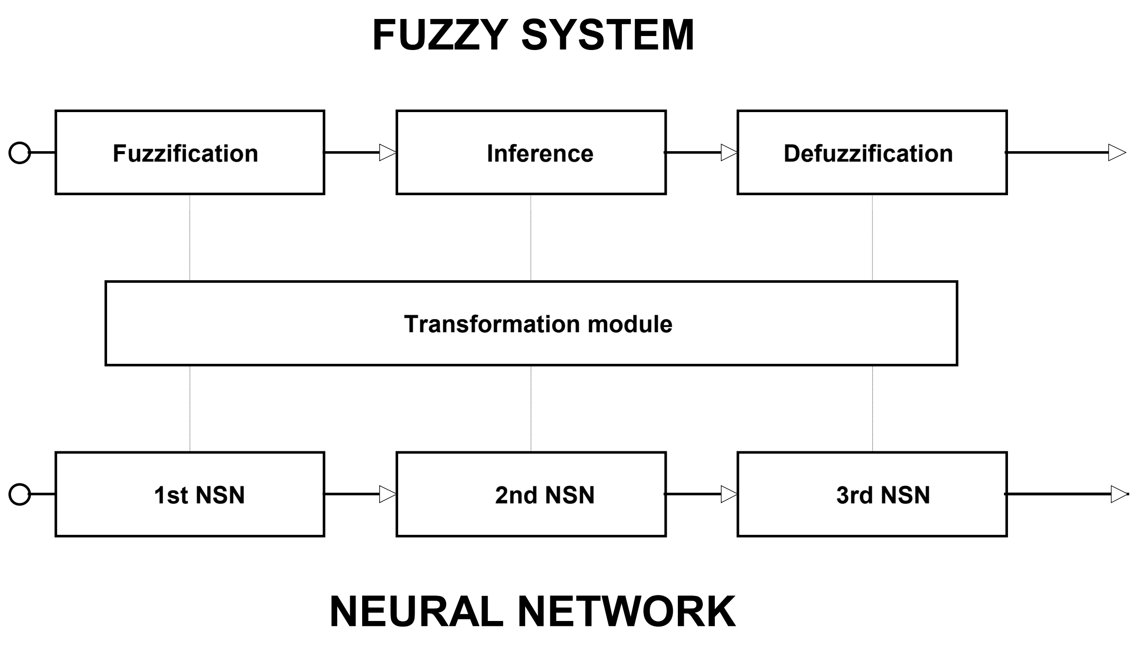
Fuzzy logic imitates and automates human behavior and provides understandable solutions. With neural networks you generate a solution using a process of learning from sample data. The advantage is the ability of the network to learn, i.e. its ability to adapt to changed behavior and new situations. To make use of the advantages of both - the understandability of fuzzy systems and the learning capability of neural networks - the two techniques are combined. The result is a neurofuzzy system. The combination results from converting a fuzzy system into a neural network and vice versa. There are two typical procedures for applying such neurofuzzy systems:

1. There is a problem, for which you can only design a bad fuzzy solution if you do not have sufficient knowledge of the problem. The fuzzy system obtained in this way is transformed into a neural network that is optimized using example data. The optimized network can either be used directly or transformed back into a fuzzy system.
2. Using a neural network, you generate a solution from a collection of sample data that, for example, identifies an unknown process behavior. If you have used a suitable network structure, you can transform the neural network into a fuzzy system. The fuzzy system - or more precisely, its rule base - can be interpreted, and in this way you can obtain an understandable description of the problem, e.g. the process behavior.

3.2 Neurofuzzy Networks (NFN)

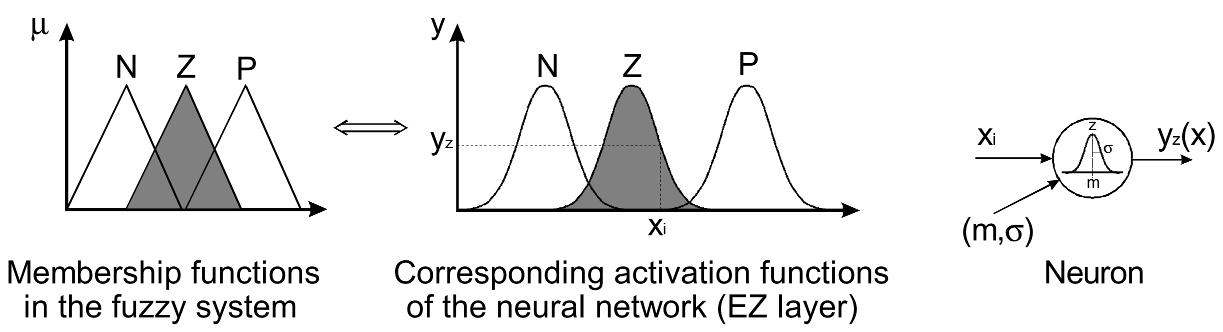
To be able to make use of the advantages of neural networks and fuzzy systems in one project, you require a system that processes fuzzy membership functions and the rule base of the fuzzy model and can determine knowledge from sample data using neurons capable of learning.

You can solve this problem by using a special neural network, called the neurofuzzy network (NFN). It consists of three neural subnetworks (NSN) that emulate the three subtasks - fuzzification, inference and defuzzification - of a fuzzy system.



1) Fuzzification in the 1st NSN

   Fuzzification of the non-fuzzy input variables is implemented by a layer of neurons with activation functions similar to those of an RBF. One neuron is assigned to each membership function of the input variables.


μ
N Z P
y
N Z P
yz
xi
xi
yz(x)
z
σ
m
(m,σ)
Membership functions in the fuzzy system
Corresponding activation functions of the neural network (EZ layer)
Neuron

The EZ layer of the 1st NSN has m neurons, where:

$$m=\sum_{i=1}^{a} n_i$$

a = number of inputs,

n_i = number of membership functions of the i^{th} input.

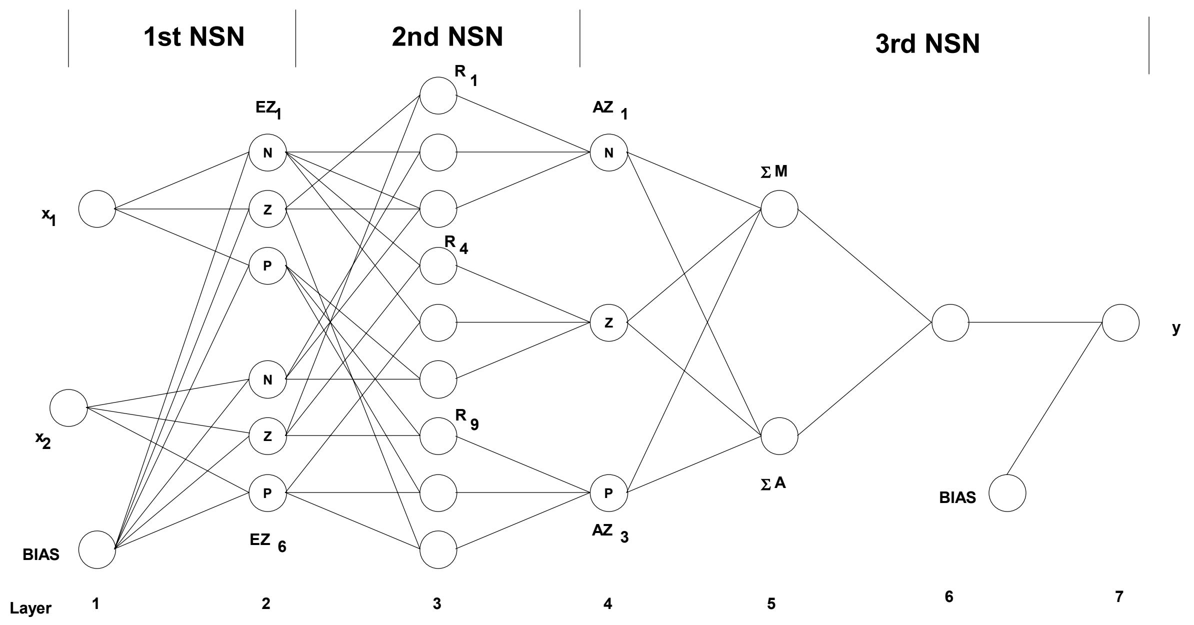
2) Inference in the 2nd NSN

The rule base of the fuzzy system is in the 2nd NSN and one neuron is assigned to each rule. The IF part of a fuzzy rules is implemented by the first neuron layer in the 2nd NSN and the THEN part of a fuzzy rule is implemented in a second neuron layer. The number of neurons in the second layer is equivalent to the number of membership functions of the output variables.

3) Defuzzification in the 3rd NSN

The output values of the system in the 3rd NSN are implemented by the standard defuzzification method with MAX-PROD inference followed by centroid calculation.

This results in a fuzzy-neuro topology like that shown in the following figure.



The system illustrated above has the input signals x_1 and x_2. Three membership functions (N-negative, Z-zero, P-positive) are assigned to each input signal. Three membership functions (N-negative, Z-zero, P-positive) are also assumed for the output signal y. Because of its neural network structure the system is capable of learning (an advantage of neural systems) and because of its fuzzy-like topology it is possible to recreate the processing steps.

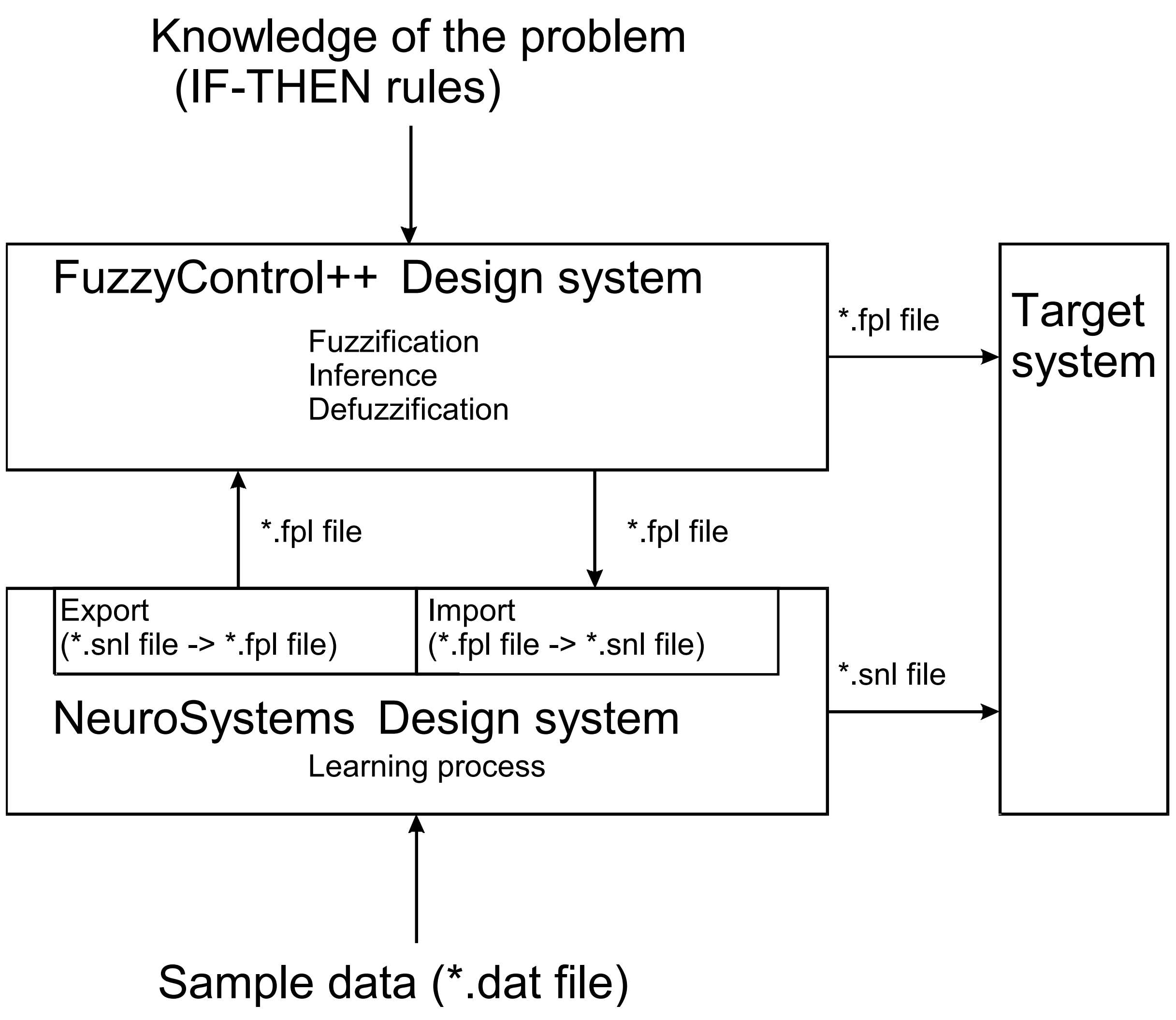
For example, a rule R4

IF x_1 is equal to N AND x_2 is equal to Z, THEN y is equal to Z

causes activation of the neuron EZ_1 (x_1; N) and neuron EZ_5 (x_2; Z) in the first NSN, if the non-fuzzy values of x_1 are within membership function "N" and x_2 within membership function "Z". In the second NSN, the neuron R_4 can form the AND combination expressed in the rule because both the inputs are active and therefore activate neuron AZ_2, which invokes the THEN condition y = Z. Defuzzification leading to a non-fuzzy output value y is performed by quotient calculation (6th layer) via the "moment" neuron (M) and the "area" neuron (A) in the 5th layer.

With this defined structure it is only possible to vary the numbers of membership functions with respect to the input and output signals. In *NEUROSYSTEMS*, therefore, you are only prompted to enter these quantities when defining the network structure *NFN*.

You can record the results of your neural or fuzzy structure development in an *.snl or *.fpl file. You can only transform the parameters from one system to another in the program *NEUROSYSTEMS* using the menu commands FILE / EXPORT or FILE / IMPORT. The following diagram illustrates how the two programs *FUZZYCONTROL++* and *NEUROSYSTEMS* work together as a "neurofuzzy system".


Knowledge of the problem
(IF-THEN rules)
FuzzyControl++ Design system
Fuzzification
Inference
Defuzzification
*.fpl file
Target system
*.fpl file
*.fpl file
Export
(*.snl file -> *.fpl file)
Import
(*.fpl file -> *.snl file)
*.snl file
NeuroSystems Design system
Learning process
Sample data (*.dat file)

3.3 Applications

For a closed neurofuzzy system, consisting of parts from *NEUROSYSTEMS* and *FUZZYCONTROL++*, two typical applications are presented that demonstrate the advantage of using the two complementary programs together:

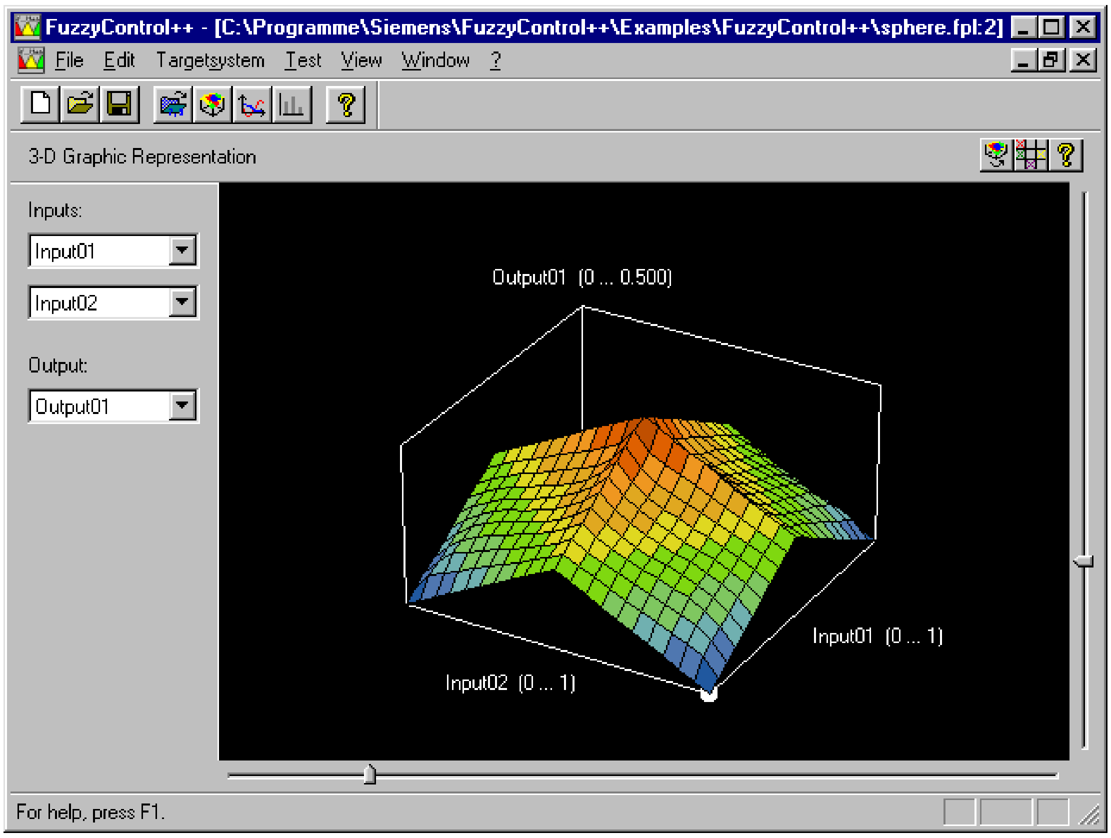
<u>1st application example</u>: A fuzzy model which is not yet sufficiently precise is optimized with a neural network.

The demonstration example is the visual display of a hemisphere in a Cartesian coordinate system (3-D display), where

Two input signals	$x \rightarrow$ Input 00	(0 to 1)	
	$y \rightarrow$ Input 01	(0 to 1)	and
One output signal	$z \rightarrow$ Output 00	(0 to 0.5)	

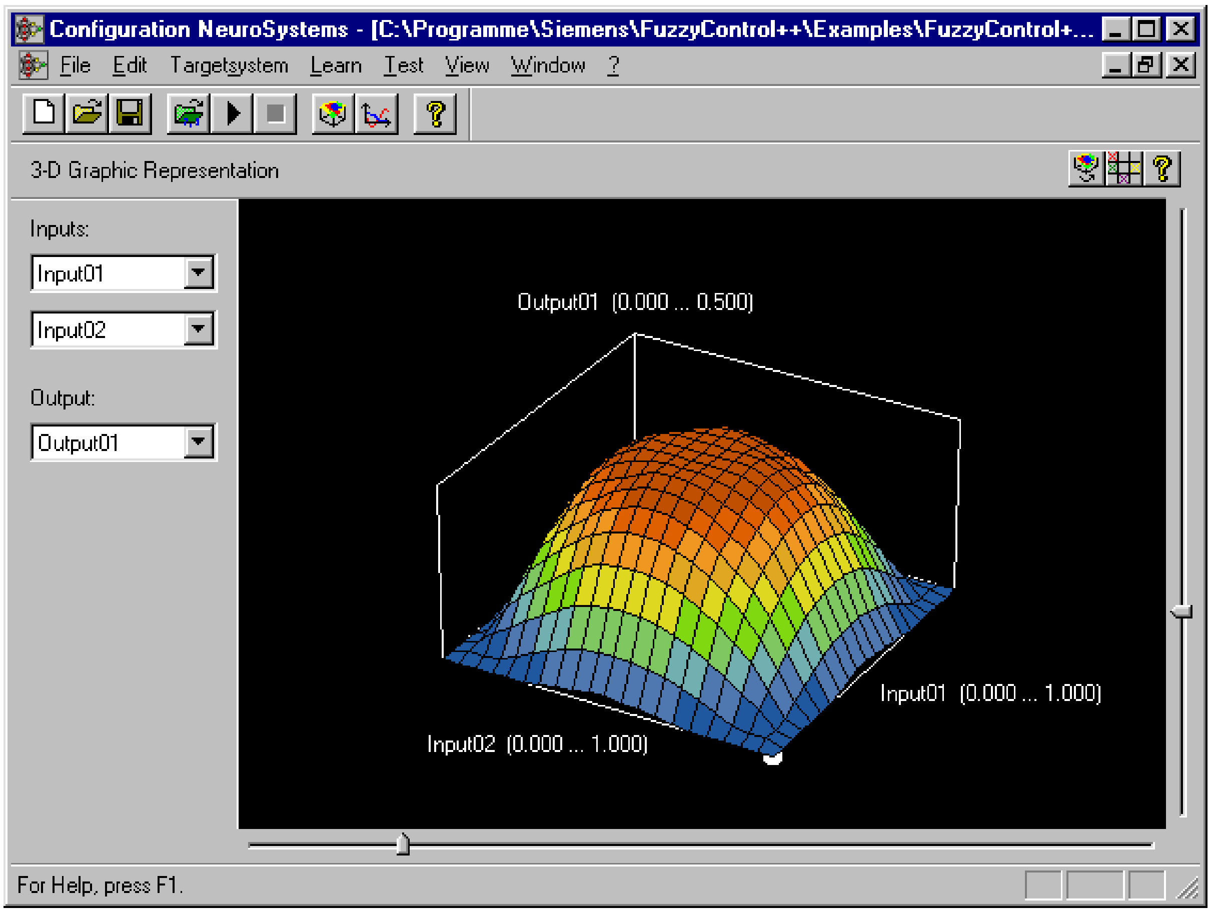
form the sphere equation $x^2 + y^2 + z^2 = r^2$ (radius $r = 0.5$).

With the fuzzy rule base you can define a few points unambiguously, e.g. the center of the sphere, the corner points of the x-y surface. Overall description as a fuzzy model by membership function and rule base is difficult, as the following diagram shows.

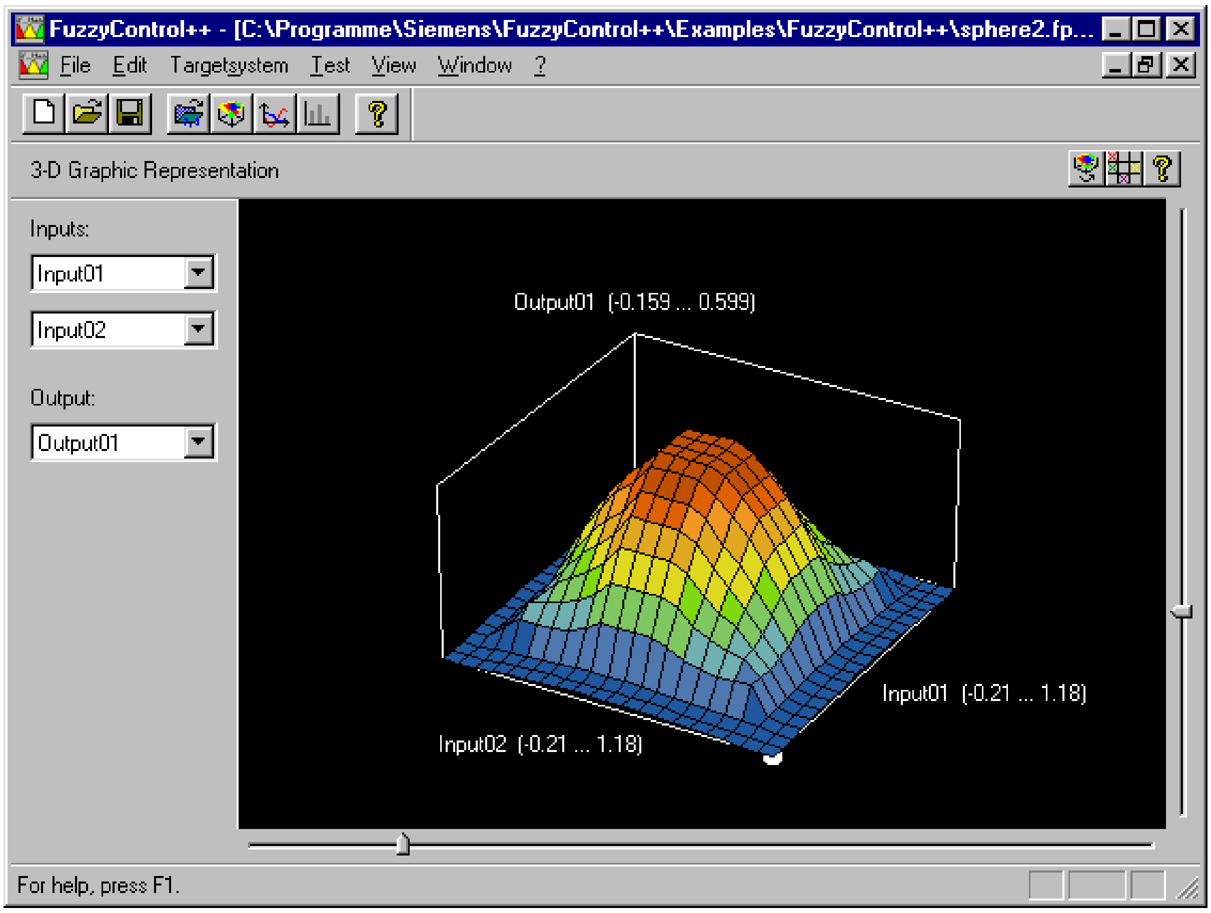

FuzzyControl++ - [C:\Programme\Siemens\FuzzyControl++\Examples\FuzzyControl++\sphere.fpl:2]
File Edit Targetsystem Test View Window ?
3-D Graphic Representation
Inputs:
Input01
Input02
Output:
Output01
Output01 (0 ... 0.500)
Input01 (0 ... 1)
Input02 (0 ... 1)
For help, press F1.

With symmetrical formation of the membership functions, as shown in this model (stored in the file "sphere.fpl"), you inevitably obtain an imprecise image of this hemisphere.

By conversion of the fuzzy parameters into those of a neural network and after a learning process using the learning data from the "sphere.dat" file (calculated by the sphere equation), the result is an improved model. (See next page for figure.)


Configuration NeuroSystems - [C:\Programme\Siemens\FuzzyControl++\Examples\FuzzyControl+...
File Edit Targetsystem Learn Test View Window ?
3-D Graphic Representation
Inputs:
Input01
Input02
Output:
Output01
Output01 (0.000 ... 0.500)
Input01 (0.000 ... 1.000)
Input02 (0.000 ... 1.000)
For Help, press F1.

Transformation back to the fuzzy system allows you to look at the rule base and the shape of the membership functions. This provides you with more information about the process for process identification.


FuzzyControl++ - [C:\Programme\Siemens\FuzzyControl++\Examples\FuzzyControl++\sphere2.fp...
File Edit Targetsystem Test View Window ?
3-D Graphic Representation
Inputs:
Input01
Input02
Output:
Output01
Output01 [-0.159 ... 0.599]
Input01 [-0.21 ... 1.18]
Input02 [-0.21 ... 1.18]
For help, press F1.

Because the membership functions are linear, a visual representation of the fuzzy model appears to be somewhat more "straight-edged" than the neural network in this example.

If you vary the parameters (number of membership functions) systematically and run learning processes, you can achieve further optimization.

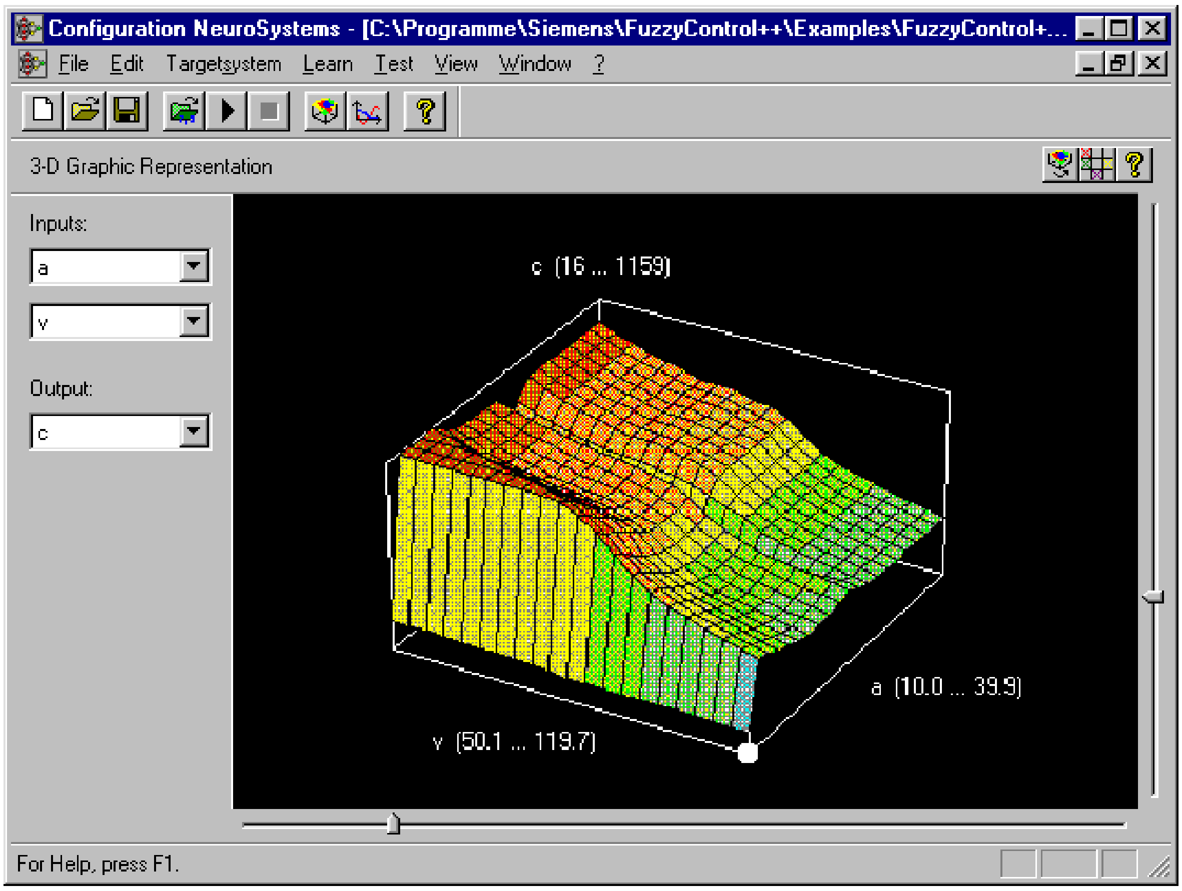
<u>2nd application example</u>: A process about which too little knowledge exists for a fuzzy model but for which series of measured data is available.

Let us take the example of coat thickness control to demonstrate the conversion process from a neural network to fuzzy model. The following measured data is available from the process

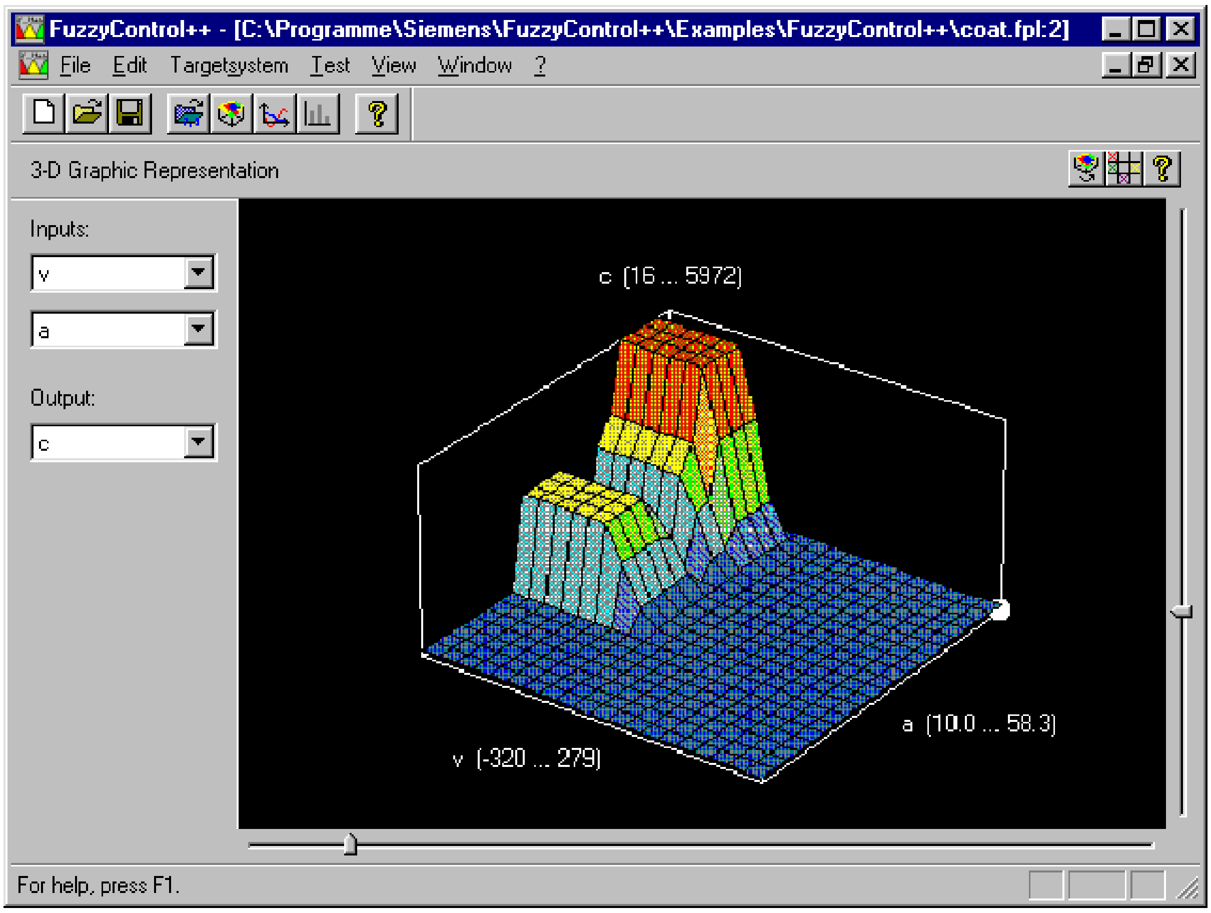
Three input signals	v - Tape velocity
	p - Air pressure
	a - Distance between the nozzle and the tape
One output signal	c - Coat layer thickness

in the file "coat.dat". Because the technical and technological conditions are complicated, it is very difficult, if not completely impossible, to establish a fuzzy model.

Training a neural network would seem the obvious solution here. The freely selectable parameters are the number of membership functions or the number of activation functions and therefore the number of neurons in the 2nd layer of the 1st NSN. The result of the learning process is shown in the following figure:


Configuration NeuroSystems - [C:\Programme\Siemens\FuzzyControl++\Examples\FuzzyControl+...
File Edit Targetsystem Learn Test View Window ?
3-D Graphic Representation
Inputs:
a
v
Output:
c
c [16 ... 1159]
a [10.0 ... 39.9]
v [50.1 ... 119.7]
For Help, press F1.

After conversion of the data file *.snl → *.fpl and transfer to the *FUZZYCONTROL++* , you can see the visual representation (following figure) and look at the rule base and the membership functions. In this way, you can also obtain information about the process which you could not derive directly from the series of measurements.



PART II:

FUZZYCONTROL++

SHORT INSTRUCTIONS

Using the *FUZZYCONTROL++* configuration tool involves two main operations:

- A fuzzy system meeting the bare requirements must be created as a basis. Subsequent changes to the fuzzy system are still possible.
- It is also necessary to establish a connection with the targetsystem and make the fuzzy system known to it.

4 System Requirements and Installation

The *FUZZYCONTROL++* configuration tool consists of a Windows program (configuration tool) and different runtime modules (function and data blocks for SIMATIC S7-300, SIMATIC S7-400 and runtime library for SIMATIC WinCC). The configuration tool is used to configure and generate fuzzy systems. The runtime modules can then execute these generated systems during runtime operation on the corresponding target platform.

There are runtime-modules for:

SIMATIC S7

SIMATIC CFC (optional)

SIMATIC WinCC

4.1 System Requirements

- Hardware requirements:

Configuration tool:	PC or PG with 80486 processor (or higher), at least 16 MB memory (RAM), at least 16 MB free hard disk space
S7 function blocks:	S7-300, CPU 314 or better, or S7-400 MPI interface for the PC The PB (PROFIBUS), TPC/IP and IE (Industrial Ethernet) bus systems are optionally supported.
CFC function blocks:	S7-400 MPI interface for the PC The PB, TCP/IP and IE bus systems are supported optionally

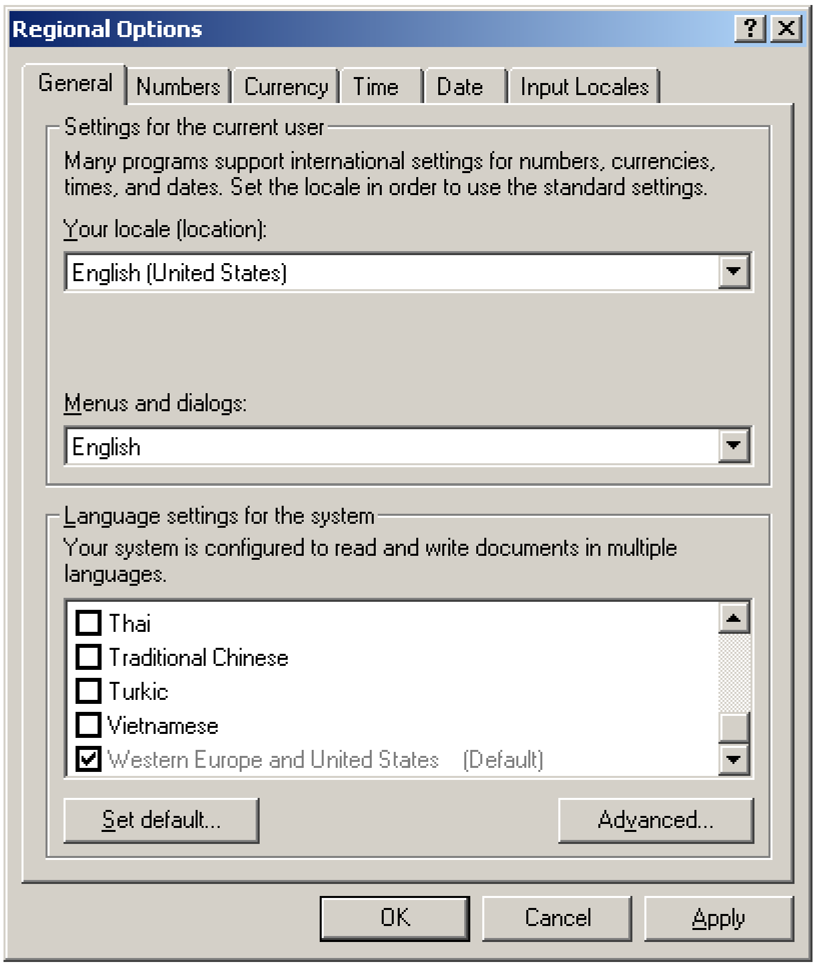
- Software requirements:

Configuration tool:	Windows 2000 or Windows XP
S7 and CFC function blocks:	For linking: SIMATIC SOFTNET S7 / PB Optionally – SIMATIC SOFTNET S7 IE with TCP/IP or IE connection

WinCC object: SIMATIC WinCC from version 3.1

4.2 Installation of *FUZZYCONTROL++*

1. Insert the *FUZZYCONTROL++* Setup CD.
2. run ***setup.exe***.
3. The setup program guides you through installation. Just follow the instructions on the screen.
4. To achieve optimum display quality on the screen, we recommend setting the Windows "Display Properties" to "small fonts"! The screen resolution should be at least 800x600 pixels.
5. The FuzzyControl++.exe installed is bilingual. The language in the *FUZZYCONTROL++* menus and dialogs is automatically set depending on your Windows system settings (Start – Settings – Control Panel – Regional Options).



Settings for the current user

Many programs support international settings for numbers, currencies, times, and dates. Set the locale in order to use the standard settings.

Your locale (location): English (United States)

Menus and dialogs: English

Language settings for the system

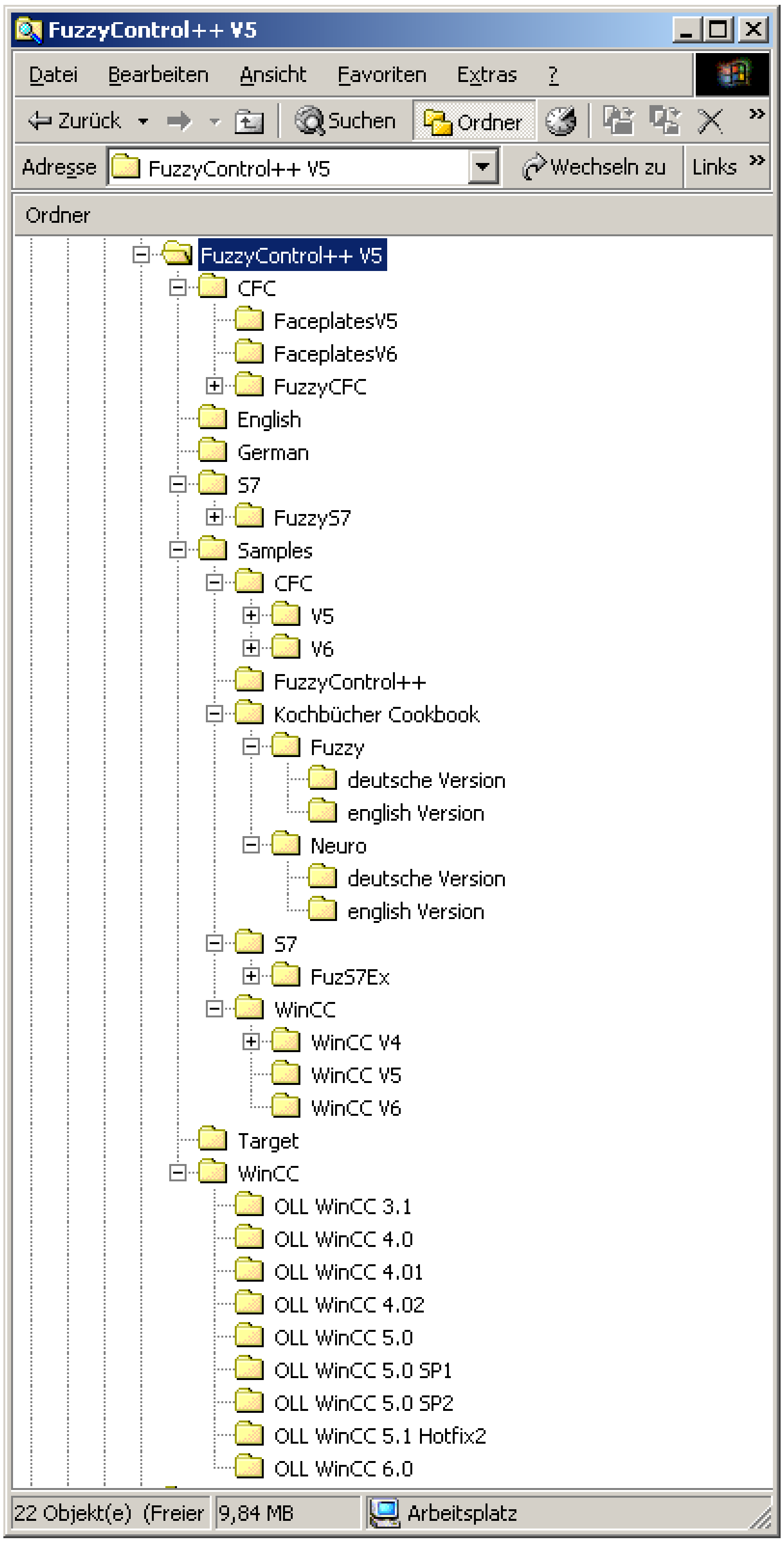
Your system is configured to read and write documents in multiple languages.

- [ ] Thai
- [ ] Traditional Chinese
- [ ] Turkic
- [ ] Vietnamese
- [✓] Western Europe and United States (Default)

After installation not only the configuration tool itself but also the following components are available for use in the relevant directories of *FUZZYCONTROL++*:

- SIMATIC S7 modules for S7 applications (**S7 Catalog**)
- SIMATIC PCS7 modules and Picture components for SIMATIC PCS7 applications (**CFC Catalog**)
- SIMATIC WinCC-OLLs from SIMATIC WinCC version 3.1 onwards (**WinCC Catalog**)
- Targetsystem driver (**Target Catalog**)
- SIMATIC S7 - example project (**Catalog Samples\S7**)
- SIMATIC CFC - example projects (**Catalog Samples\CFC**)
- SIMATIC WinCC - example project (**Catalog Samples\WinCC**)
- *FUZZYCONTROL++* example projects (**Catalog Samples\FuzzyControl++**)
- Cookbook with applications (**Catalog Samples\Cookbook**)
- Handbook, Help (**English Catalog, German Catalog**)

You can uninstall the program and the installed files using the Windows utilities. Instructions can be found in the Windows help program and the Windows User Manual.


FuzzyControl++ V5
Datei Bearbeiten Ansicht Favoriten Extras ?
Zurück
Suchen
Ordner
Adresse FuzzyControl++ V5
Wechseln zu
Links
Ordner

```
FuzzyControl++ V5
    CFC
        FaceplatesV5
        FaceplatesV6
        FuzzyCFC
    English
    German
    S7
        FuzzyS7
    Samples
        CFC
            V5
            V6
        FuzzyControl++
        Kochbücher Cookbook
            Fuzzy
                deutsche Version
                english Version
            Neuro
                deutsche Version
                english Version
        S7
            FuzS7Ex
        WinCC
            WinCC V4
            WinCC V5
            WinCC V6
    Target
    WinCC
        OLL WinCC 3.1
        OLL WinCC 4.0
        OLL WinCC 4.01
        OLL WinCC 4.02
        OLL WinCC 5.0
        OLL WinCC 5.0 SP1
        OLL WinCC 5.0 SP2
        OLL WinCC 5.1 Hotfix2
        OLL WinCC 6.0
```

22 Objekt(e) (Freier 9,84 MB Arbeitsplatz

5 Quick Guide To Creating A Fuzzy System

The starting point in processing a project is the problem to be solved (with the targetsystem). A *project* is a fuzzy system specified by the choice of targetsystem, its inputs and outputs and its rule base. To edit a project, you can either open an existing project or create a new project. For an existing project you can load its file with the extension **.fpl (FUZZY PROGRAMMING LANGUAGE)* or **.fcl* (*FUZZY CONTROL LANGUAGE)* with *Open...* in the *File* menu*.* A new project is automatically given the project name "New". The first time you save the project (*File/Save as...*), you can assign any name to it.

Let us take a very simple example in which a control must adjust a valve depending on the temperature and pressure. This control is to be implemented in a fuzzy system and a program for this purpose is to be generated and transferred in the control unit, which is the targetsystem. With the *FUZZYCONTROL++* tool, you can create the required fuzzy system.

This quick guide provides an introduction to editing a project. You can work through a typical project if you perform the steps (indented text marked as Example) with the *FUZZYCONTROL++* tool on the computer.

The emphasis is not on solving a practical problem but on explaining the principles using an example, which is easy to understand.

5.1 Configuration Of The Fuzzy System

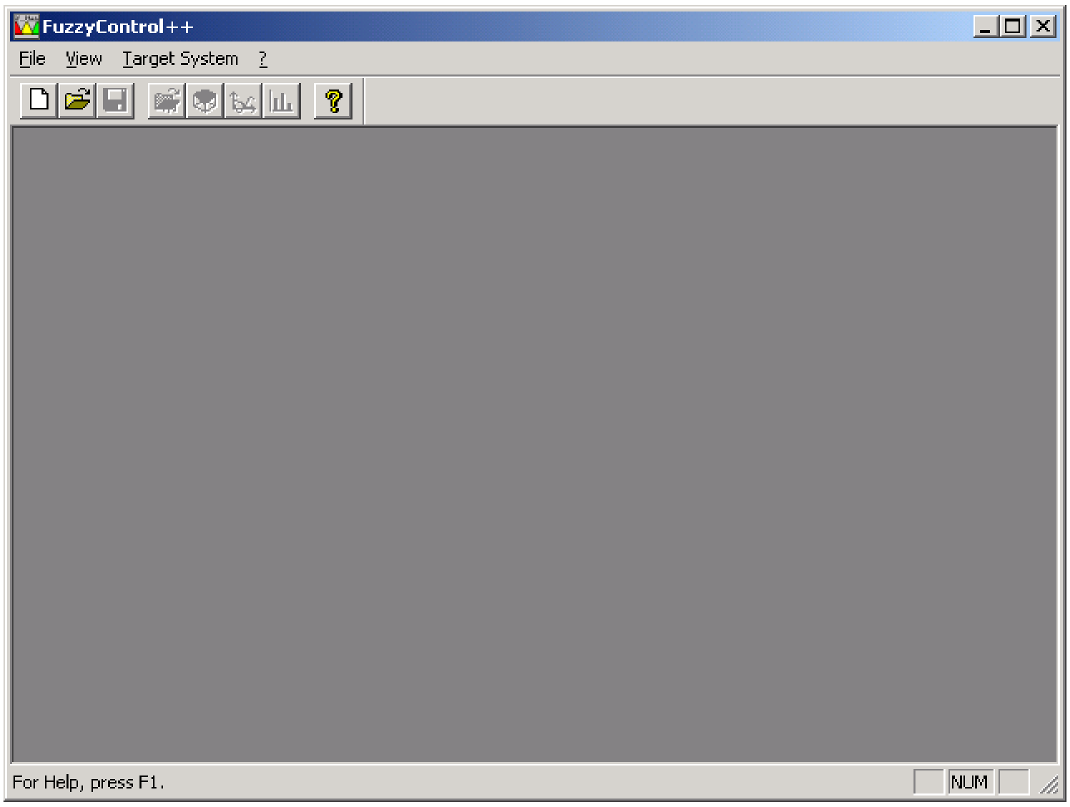
Configuration of the fuzzy system includes defining the targetsystem and the number of inputs and outputs of the fuzzy system. In addition, a standardization must be prepared by the program by specifying minimum and maximum values for the inputs and outputs.

The targetsystem can also be indicated or changed at a later time. To insert or change targetsystems, see the Targetsystem chapter.

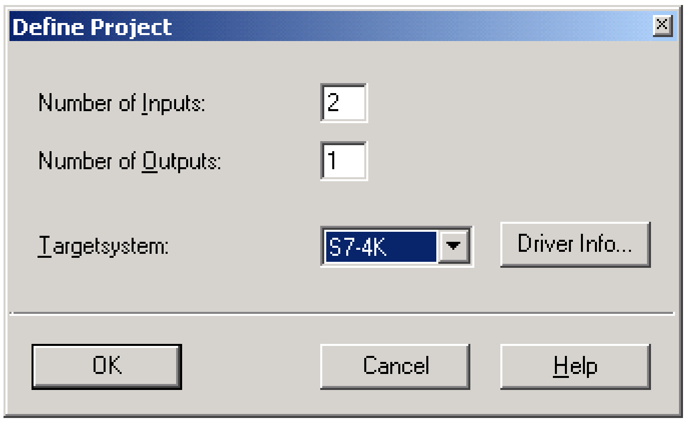
5.1.1 Defining the number of inputs and outputs as well as the targetsystem

Example:

After you have started the *FUZZYCONTROL++* program, a basic window with the title " *FUZZYCONTROL++* " is displayed. This is the working window of *FUZZYCONTROL++*. The toolbar (top) and the status bar (bottom) are displayed in all the windows and assist you operating the program.



Run the *New* command in the *File* menu displayed in the window shown above. First of all, a dialog box with the name *Define Project* in the title bar appears in which you can enter the external architecture.



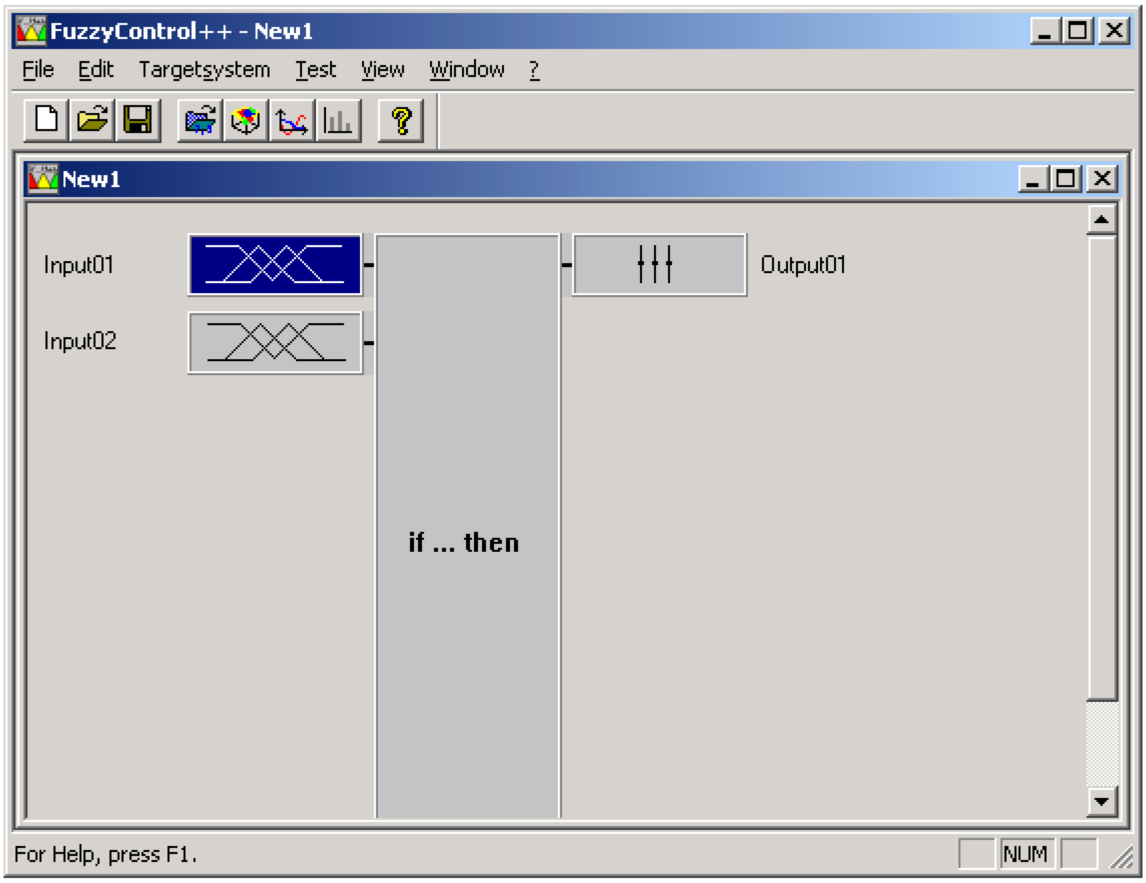
Number of Inputs: 2
Number of Outputs: 1
Targetsystem: S7-4K

The default number of inputs and outputs displayed are those chosen for the valve example. As the targetsystem we could, for example, select SIMATIC S7-4K. If you click OK, the external architecture configuration is completed. The project is now given the name *New1*.

Note: After you have clicked *OK*, you will see all the menus that are available in the working window.

You have not set the number of inputs and outputs permanently. You can insert or delete inputs and outputs later on with the *Edit* menu. See Part III of the Manual for more details.

After you have completed the external configuration, a window with a block diagram of the fuzzy system is displayed. This window forms the basis for editing the project (project window). It shows a symbolic representation of the rule base (if...then block) and its inputs and outputs (in this example the default number of inputs and outputs). The provisional project name *New1* is used for the project window.

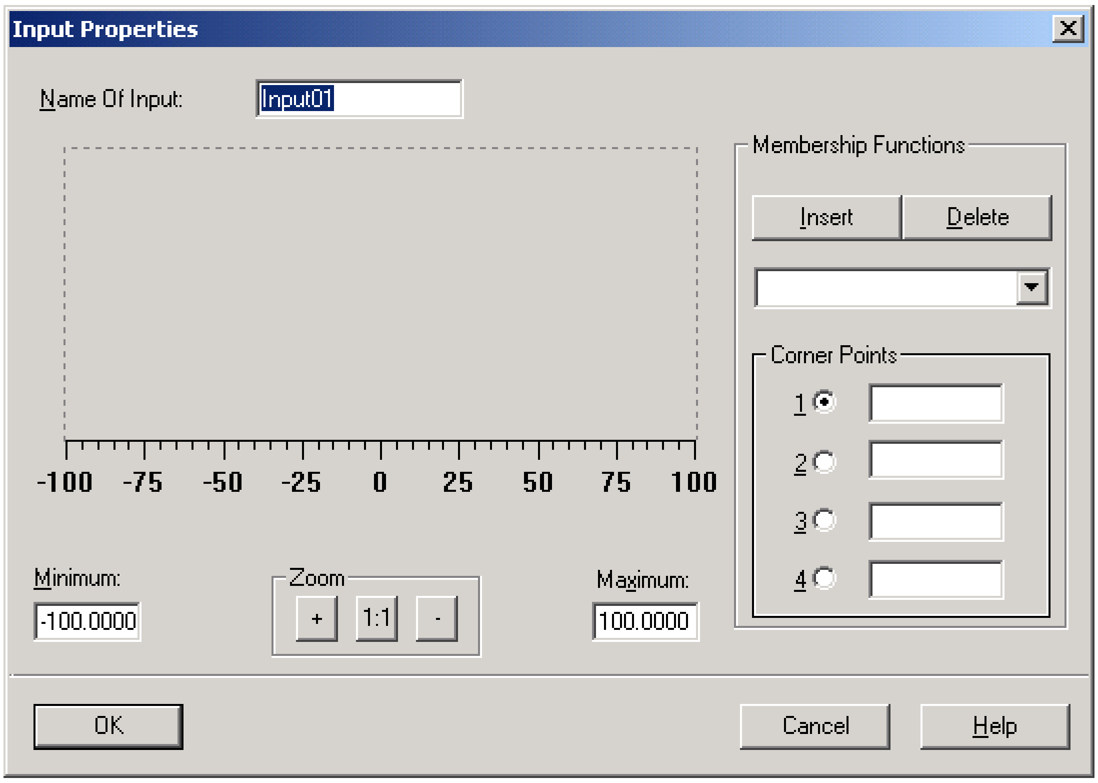


5.1.2 Editing the inputs and outputs

When you edit the project, you can assign meaningful names to the inputs and outputs. You can also normalize the inputs and outputs, if necessary. This is what we are now going to do.

- **Edit Input01:**

We are in project window *New1*. If you double-click the uppermost input button (*input01*) with the left mouse button, the *Input Properties* dialog box for this input is displayed. The name also appears in the entry field Change this name to "Temp". The input should now be normalized to a temperature range of 0-30°C. Enter "0.0" in the *Minimum* field and "30.0" in the *Maximum* field. Editing of the inputs is completed with *OK* in the *Input Properties* window.



Name Of Input: Input01

Minimum: -100.0000

Maximum: 100.0000

- **Editing input02:**

If you double-click this input button with the left mouse button, the *Input Properties* dialog box for this input appears, indicated by the input number 02. Enter the name "Pressure" and the values "50.0" and "100.0" for the *Maximum* and *Minimum* values.

- **Editing output01:**

If you double-click on the output button with the left mouse button, the *Output Properties* dialog box for that output appears, indicated by output number 01. Enter the name "valve" here and the values "-101.0" and "101.0" for the *Maximum* and *Minimum* values.

Complete editing the inputs and outputs by closing the *Output Properties* dialog box with *OK*. The block diagram of the fuzzy system (project window) is displayed again.

5.2 Membership Functions

Defining the membership functions

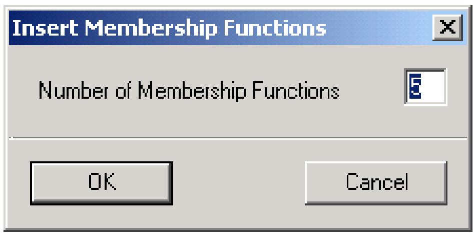
After naming the inputs and outputs and normalizing them, the next step is to define the membership functions.

To do that you must define the number of membership functions per input and output and their shape. The standard form is the trapezoid. Its four corner points are numbered 1 to 4 from left to right. Assignment of process values to these points determines the special shape of the membership function.

Example:

- **Membership functions for the "Temperature" input**

You define and edit membership functions in the *Input Properties* window, which you can open by double-clicking on the "Temp" input in the block diagram. Now determine is how many membership functions you want to define for this input. Click on the *Insert* button. with the left mouse button.


Insert Membership Functions

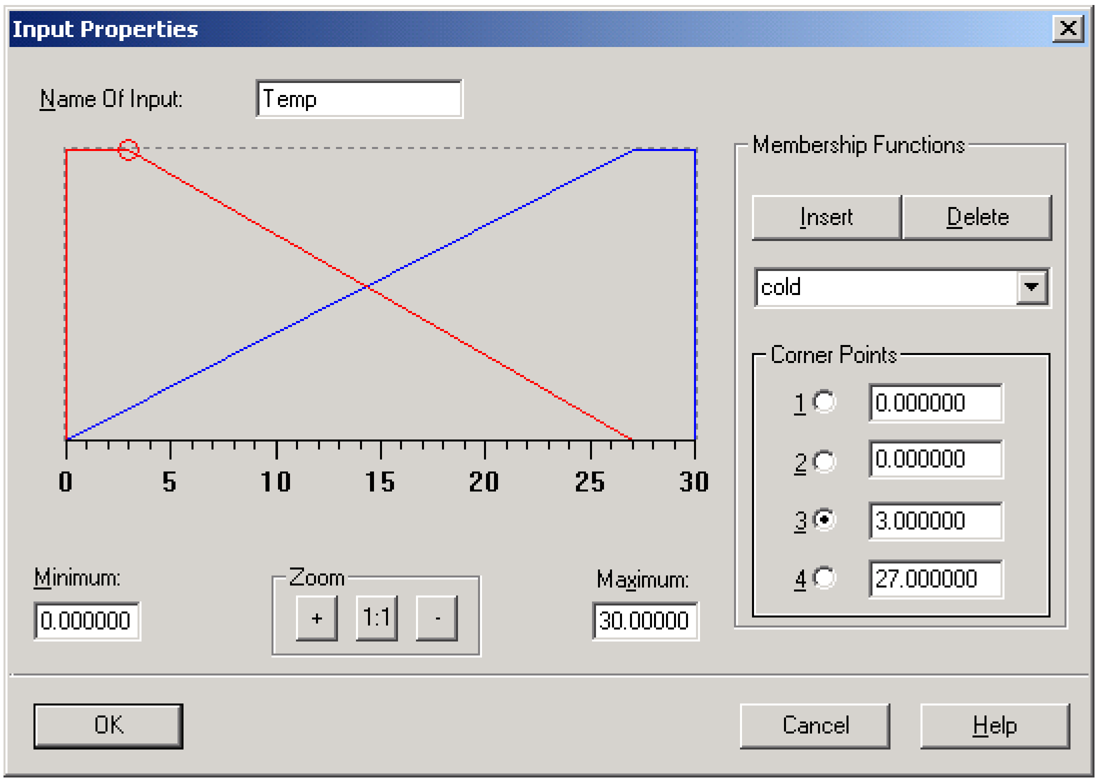
Number of Membership Functions: 5

The dialog window *Insert Membership Functions* is then opened. Enter the number 2 is in the *Number of Membership Functions* field (You can define up to seven membership functions per input). After confirming your choices with *OK*, the membership functions are entered in the diagram window.

The number of membership functions is not permanent. You can still insert or remove membership functions.

The functions entered are first displayed as triangles, in this example as edge triangles. The triangles are entered in such a way that they overlap to equal degrees.

After you have inserted the membership functions, you can assign your own names to them. These names can be up to 7 characters long, but may not contain special characters. They may not begin with a number either. To change a name, select the membership function in question from the list box below the two buttons. You can then change the name. Rename the membership function "negative" as "cold" and the membership function "positive" as "hot".


Input Properties

Name Of Input: Temp

Membership Functions: cold

Corner Points:
1: 0.000000
2: 0.000000
3 (selected): 3.000000
4: 27.000000

Minimum: 0.000000

Maximum: 30.00000

The next step is to change the points of the membership function. Firstly, select the membership function for which you want to change the points. The selected membership function appears as a red line in the diagram. Select the value of a point to change it. Click in the *Corner Points* frame on one of the buttons *1, 2, 3* or *4*. You can then change the value of the point either by entering a value in the entry field next to the button or by clicking and dragging the red circle in the diagram. In the figure above, the example values have been entered.

Complete editing the membership function by confirming your entries with *OK*.

Note: It's necessary for the values of the points that: $P_1 \le P_2 \le P_3 \le P_4$!

- **Membership functions for the "Pressure" input**

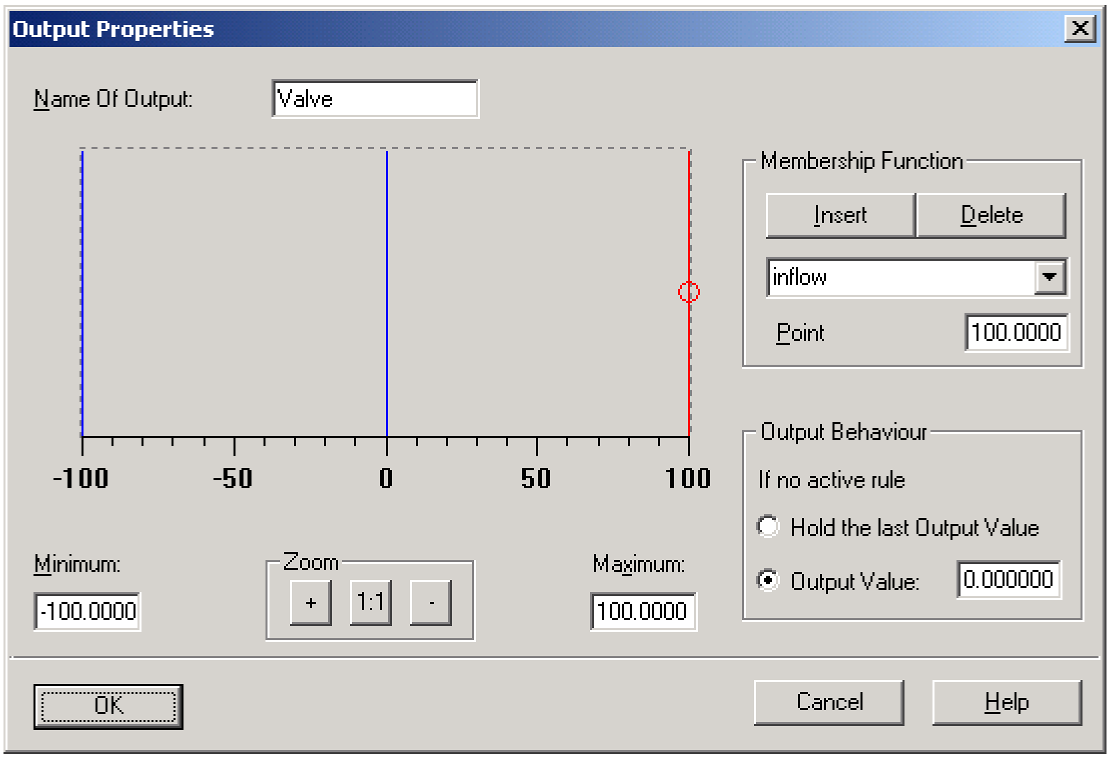
Open the *Input Properties* dialog box by double-clicking on the "Pressure" input. Then open the *Insert Membership Functions* dialog box using the *Insert* button. Two membership functions are inserted for this input. After confirmation with *OK* these membership functions are displayed as edge triangles. Rename the membership function "negative" to "low" and enter the values "50.0", "50.0", "55.0" and "95.0" for the points. Rename the membership function "positive" to "high" and enter the values "55.0", "95.0", "100.0" and "100.0" for the points. You can complete editing by confirming your entries with *OK*.

- **Membership function for the "Valve" output**

Open the *Output Properties* dialog box by double-clicking on the "Valve" output. *Insert* three membership functions using the *Insert Membership Functions* dialog box which is opened by clicking on the Insert *button.*

Note: You can define up to nine membership functions for each output.

However, unlike the inputs, the outputs are inserted as singletons and not as triangles. So each membership function only requires 1 point to define its position. The values of the points are the ones required for this example and do not need to be changed. Rename the membership functions "negative", "zero" and "positive" to "outflow", "closed" and "inflow".



Name Of Output: Valve

Membership Function: inflow

Point: 100.0000

Minimum: -100.0000

Maximum: 100.0000

Output Behaviour

If no active rule

Hold the last Output Value

[✓ Output Value:] 0.000000

The final step is to define the behavior of the output for the case in which the rule is not active. Select *Output Value* and set the value to "0.0". That means that the valve is always closed if no rule is active.

Complete editing the output with *OK* and the *Output Properties* dialog box is closed. The block diagram of the fuzzy system reappears.

5.2.1 Rule Matrix

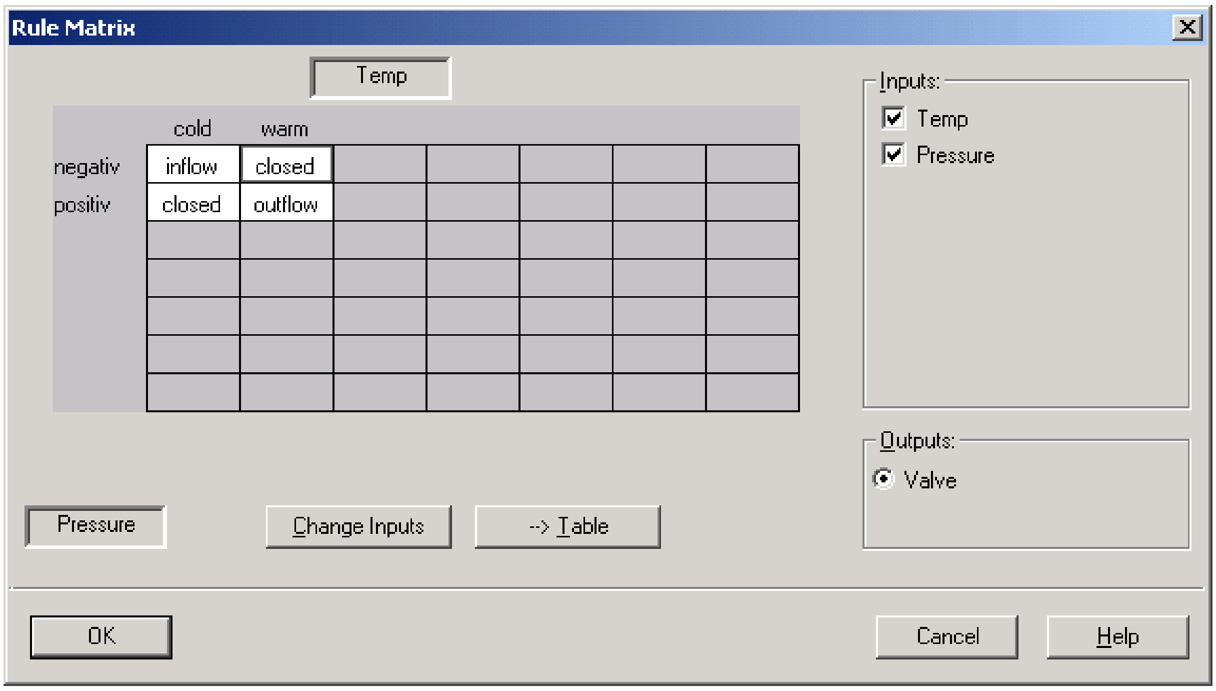
Open the *Rule Table* dialog box by double-clicking on the if...then block with the left mouse button. This dialog box is used to set up and edit the rules for the fuzzy system. Alternatively, you can edit the rules in the *Rule Matrix* dialog box (opened by clicking on the if...then block with the right mouse button and then selecting *Rule Matrix*). It is clearer to work with the rule matrix for this fuzzy system because the matrix only works with two inputs and one output. The rule table is preferable for smaller systems with a few rules or for large systems with several outputs. In the rule table, a column always corresponds to one rule. In large systems whose rule base is near to the maximum limits, you can reduce the number of rules by assigning one rule to several outputs.

Note: It is always possible to convert the rule matrix to a rule table, but it is not always possible convert a rule table to a rule matrix.

Example: The example system is to contain the four following rules:

1. If Temperature "cold" and Pressure "low" then Valve "inflow"
2. If Temperature "cold" and Pressure "high" then Valve "closed"
3. If Temperature "hot" and Pressure "low" then Valve "closed"
4. If Temperature "hot" and Pressure "high" then Valve "outflow"

It is easier to use the rule matrix to enter these rules. Open the *Rule Matrix* dialog box by clicking on the if...then block with the right mouse button and then select *Rule Matrix*. In the right hand side of the window, "Temp" and "Pressure" are already selected as the inputs and "Valve" is already selected as the output for editing in the matrix (on the left-hand side of the window). The rules are entered so that for every combination of inputs, one of the linguistic values is selected from the output list box. In the end, the matrix should appear as it does in the figure below.



	cold	warm
negativ	inflow	closed
positiv	closed	outflow

Complete entering the rules by confirming your entries with *OK*.

After you have completed entering the rules, the fuzzy system is complete and can now be processed by the targetsystem.

6 Communication Between *FUZZYCONTROL++* And the SIMATIC Family

The fuzzy system is created with *FUZZYCONTROL++* and loaded into an S7-CPU for execution, or processed within SIMATIC WinCC.

To communicate between the configuration tool and the SIMATIC S7, the *FUZZYCONTROL++* configuration tool, the SOFTNET product from SIMATIC NET, as well as an MPI card (Multi Point Interface) at least are required on the PC side. **CP5511 is not supported.**

On the S7 side your require the corresponding CPU (e.g. CPU 314-1 or CPU 412-1) as well as the corresponding function and data blocks which will load the fuzzy system.

6.1 SIMATIC-S7 Function Blocks

6.1.1 General

FUZZYCONTROL++ S7 blocks can be used for process control in the same way as other SIMATIC S7 software components in the same programmable controllers – and also in conjunction with the functionality of other blocks.

The algorithms for a particular fuzzy application are calculated in the processor (CPU) of the SIMATIC S7 programmable controller and, more precisely, after an unconditional call by a user program or cyclically at time-controlled intervals. The results are stored in the associated instance DB and forwarded to the peripherals accordingly.

When being used by the SIMATIC S7 function blocks, each fuzzy application is represented by its DB (instance DB), which is described implicitly by the *FUZZYCONTROL++* configuration tool.

Knowledge of SIEMENS STEP 7 software is necessary for the interaction with the SIMATIC S7 blocks.

6.1.2 Library Contents

For the SIMATIC S7 series of controllers, the following function blocks for FUZZYCONTROL++ exist:

FB 30:	S7 function block for S7-300/400	(1206 bytes)
FB 31:	S7 function block for S7-400	(1206 bytes)
DB 30:	Data block for FB 30	(4132 bytes)
DB 31:	Data block for FB 31	(20516 bytes)

For the SIMATIC S7-300 a 4Kbyte data block is available and for the SIMATIC S7-400 there is a larger 20Kbyte data block available. For both DBs, an FB exists which is designed for the DB. It is possible to implement several fuzzy systems with one DB for each system and a common FB for all systems.

6.1.3 Installing the Library

The data and function blocks for the S7-4K and S7-20K drivers are automatically inserted during the installation of FUZZYCONTROL++ into the FuzzyControl++V5\S7 directory in the form of a library ("FuzzyS7_Library"). If the STEP7 software has been installed beforehand, then these libraries are additionally inserted into the Siemens\Step7\S7libs directory. Additionally, the library and examples are placed in the FuzzyControl++V5 target directory during installation of FUZZYCONTROL++.

To be able to use these function blocks on the controller, you must create a project with the SIMATIC manager. Within this project, the controller used is specified and an S7 program is generated. In this program, the function blocks required are copied from the "FuzzyS7_Library".

Note: You can freely change the numbers of the function blocks - within the limits of the CPU. The presets are FB30 (for the "small" DB) and FB31 (for the "large" DB). They can be renamed using the STEP 7 software.

6.1.4 Functions of the „Fuzzy_FB_4K“ and „Fuzzy_FB_20K“ Components

The component calculates a maximum of 4 outputs from a maximum of 8 inputs using predetermined rules defined in the Fuzzy tool. You can use the projects with S7-300 CPUs and S7-400 CPUs.

Mode of Operation

If the value at the “START_STOP” input is 0, the rule base is inactive. If the value changes to non-zero, output values will be calculated by predetermined rules with aid of the inputs “INPUT1” to “INPUT8” and written to “OUTPUT1” through to “OUTPUT4”.

6.1.5 Start-up behavior

In a new fuzzy application, the FB must be entered into the empty data block first. Only when it is entered, is the data block recognised as a fuzzy data block by the configuration tool and allowed to be defined. I.e. At the beginning the component must be run at least once so that it is initialised. The component is additionally called in OB100 (automatically) and the initialization takes place in this OB during startup

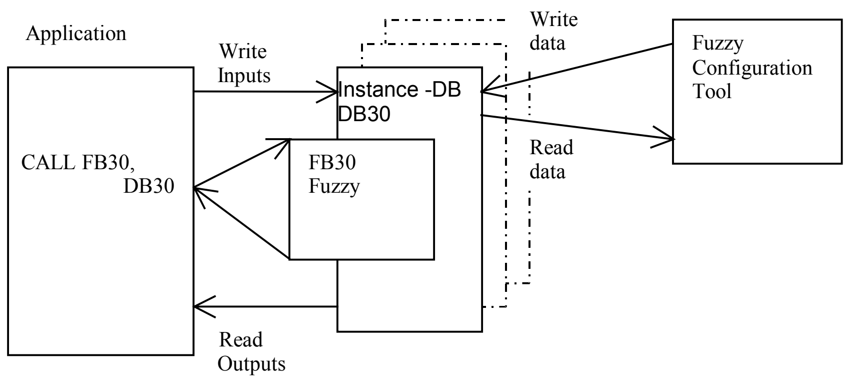
6.1.6 Component Structure

In the fuzzy **function block** (FB), all algorithms and procedures are implemented with the all the functionality that goes with high performance fuzzy application:

- Fuzzification of the inputs,
- Processing the rules,
- Defuzzification and returning the results at the outputs.

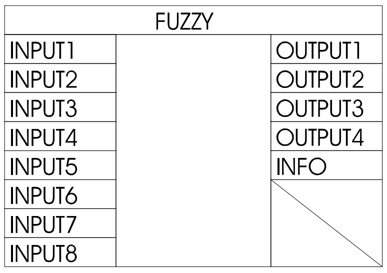
The instance **data block** (DB) in the CPU of the programmable controller sets up the interface between the function block, the configuration tool and the user. When calling the FBs, the inputs (INPUT1,…,INPUT8) must be provided with the desired values, i.e. the addresses of where the inputs are stored should be stated. After being processed by the FB, the values written to the outputs can be read out from the instance DB.

The membership functions and rules are 'implicitly' entered by the configuration tool into the instance DB. You can transfer several fuzzy applications to- and operate them on one CPU. Each application is stored in a separate DB, which can be assigned any number. It must also be noted that in the fuzzy configuration tool, the DB number, in which the values must match



6.1.7 Block diagram and the FB parameters

The fuzzy function block for S7 applications has the following block diagram:



Input parameters

The following table shows the data types and the structure of the FB input parameters.

Byte	**Parameter**	**Data type**	**Description**	**Default**
0	INPUT1	REAL	1st fuzzy application input	0.0
4	INPUT2	REAL	2nd fuzzy application input	0.0
8	INPUT3	REAL	3rd fuzzy application input	0.0
12	INPUT4	REAL	4th fuzzy application input	0.0
16	INPUT5	REAL	5th fuzzy application input	0.0
20	INPUT6	REAL	6th fuzzy application input	0.0
24	INPUT7	REAL	7th fuzzy application input	0.0
28	INPUT8	REAL	8th fuzzy application input	0.0

Output parameters

The following table shows the data types and the structure of the FB output parameters.

Byte	Parameter	Data type	Description	Default
32	OUTPUT1	REAL	1st fuzzy application output	0.0
36	OUTPUT2	REAL	2nd fuzzy application output	0.0
40	OUTPUT3	REAL	3rd fuzzy application output	0.0
44	OUTPUT4	REAL	4th fuzzy application output	0.0
48	INFO	BYTE	Process information	B#16#0

Additional parameters

In addition to the input and output parameters, the FB also has two further parameters:

Byte	Parameter	Data type	Description	Default
50	START_STOP	WORD	<> 0: Execute fuzzy application == 0: Do not execute fuzzy application	W#16#0
52...	FUZZY	BYTE	Internal field for the FB	B#16#0

6.1.8 Calling the Function block

The fuzzy function block (FB) must be called by the user. The call can be made on a cyclically (in OB1) and/or on a time-controlled program execution level (e.g. every 100ms in OB35) and it the same applies for both FBs. The FB **must** be called unconditionally. It is controlled via the START_STOP variable in the instance DB.

The call to the function block call is integrated with the help of the S7-Manager Block Editor. The call under STL is as follows:

```
CALL    FB31 , DB31
   INPUT1 :=
   INPUT2 :=
   INPUT3 :=
   INPUT4 :=
   INPUT5 :=
   INPUT6 :=
   INPUT7 :=
   INPUT8 :=
   OUTPUT1:=
   OUTPUT2:=
   OUTPUT3:=
   OUTPUT4:=
   INFO:=MB 50
```

During a call to the function block, you must specify the desired instance data block (fuzzy DB) that contains the fuzzy application which was loaded and created using the configuration tool.

You only need to enter the parameters required by the rule base. Inputs and outputs that aren't used do not need to be connected.

A minimum call with the INFO parameter would have the following listing:

```
STL
CALL FB30, DB30
(             INFO := MB30);
```

After this step, the program can now be transferred to controller and RUN mode can be switched on.

To test the fuzzy system, you can set the inputs, for example, with values via the S7 variable table and then read out the values determined by the fuzzy system at the outputs. The output *INFO* provides information about the state of the fuzzy application. This information is subdivided into three categories: No error, Warning or Error (see chapter 6.1.12).

6.1.9 External setting of the I/Os

External access to the fuzzy application in the program is possible after a data block has been created as an instance and a name has been assigned in the symbolic notation.

Example:

```
..                                    Symbolic notation of DB
..
T    "Pendulum".INPUT1
..                                    Variable in DB
L    "Pendulum".OUTPUT2
..
usw
```

6.1.10 Execution control

Execution of the fuzzy application **must** be controlled via the START_STOP variable in the data block. This variable can be controlled and read by the operator. The configuration tool also changes the START_STOP variable during the data transfer.

You can directly influence execution of the fuzzy application via the START_STOP variable:

START_STOP	**Meaning**
=W#16#0000	The fuzzy application is **not** being executed
≠W#16#0000	The fuzzy application is being executed

Note: If you do not want a fuzzy application to be executed cyclically, you can control execution with the value of the START_STOP variable: e.g. by calling the function block in OB1 and by forming a timeslice in the time-controlled OB.

6.1.11 Influence using the Configuration Tool

The execution of the fuzzy application is also controlled by the configuration tool. Before transmission of fuzzy data to the DB from the PG/PC is carried out, execution is stopped by setting the START_STOP variable to W#16#0000. After transmission, the tool enters the value not equal to W#16#0000 in the START_STOP variable, which allows execution to resume.

Example:

STL	Explanation
L 0	
T "Pendulum".START_STOP	Rule base is not processed
l 123	
T "Pendulum".START_STOP	Rule base is processed
I "Pendulum".START_STOP	
l W#16#FFFF	
==I	
= M 10.0	Change of the fuzzy application by the configuration tool

Note: The configuration tool always allows execution after a transmission even if the START_STOP variable was set to W#16#0000 by the user program before transmission.

6.1.12 Evaluating the "INFO" Parameter

The block provides information about the state of the fuzzy application via the INFO parameter. This information is subdivided into three categories: No error, Warning and Error.

- **No error**

  If execution of the fuzzy application was completed without error, the variable INFO = "B#16#00".

- **Warning**

If the START_STOP variable = W#16#0000 (fuzzy application not being executed), the INFO parameter = B#16#01.

This shows you that the outputs have not been updated, but are actually the old values.

- **Error**
  - If an instance DB of the correct length is present in the CPU but no fuzzy application has been loaded from the configuration tool, the INFO parameter = B#16#11.
  - If the length of the instance DB specified is inadequate, the INFO parameter = B#16#21.

Content (B#16#...)	Interpretation
00	No error has occurred during execution
01	Execution of the rule base is blocked by the user or the configuration tool
11	The instance DB contains no valid fuzzy rule base
21	Length of the data block is inadequate (no fuzzy DB)

Note: If an error is found or a warning occurs, the outputs are not deleted. After evaluating the INFO variable, you must decide whether the old output values are to be processed or whether a defined value should be outputted.

Caution: If the FB finds a warning or error, execution of the function block is terminated immediately.

6.1.13 Typical execution times

Execution of fuzzy functions are computationally intensive operations. The execution speed of specific fuzzy applications depends on the performance of the CPU used. The more often the CPU must calculate the output variables per unit time, the lower the number of fuzzy systems that can be installed

Processing times differ because of the dependence on the number of inputs/outputs, number of rules and the programmable controller. The following runtime measurements provide an indication of runtimes (S7-300 with CPU 314 and S7-400 with CPU 413-1):

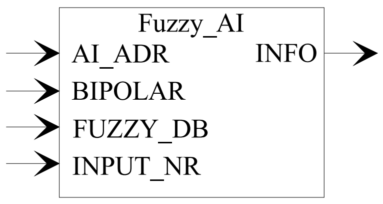
Inputs Membership functions	2 5 each	2 7 each	2 7 each	8 5 each	8 7 each	8 7 each
Rules	20	200	1000	10	100	1000
Outputs Membership functions	1 5	1 9	1 9	4 5 each	4 9 each	4 9 each
Runtime on S7-300 **Runtime on S7-400**	**approx. 13.5 ms** **approx. 1.8 ms**	**approx. 78 ms** **approx. 11 ms**	***** **approx. 73 ms**	**approx. 31 ms** **approx. 4 ms**	**approx. 180 ms** **approx. 22 ms**	***** **approx. 154 ms**

* Fuzzy system too large for S7-300

6.1.14 Reading (FC30) and Writing (FC31) Analog Values

6.1.14.1 FC30: Read Analog Value

The Fuzzy_AI function normalizes an analog input (voltage or current values) with the appropriate upper and lower limits, which are defined in the fuzzy rule base, and enters the normalized value into the instance data block. In the user program, the block must be called for each input channel of the fuzzy system that is set directly with an analog input. The user can freely assign the number of the Fuzzy_AI function (within the limits of the CPU). The default setting is FC30.



Input parameters:

Variable	Data type	Comment
AI_ADR	INT	Address of the analog input
BIPOLAR	BOOL	Bipolar or unipolar measurement
FUZZY_DB	BLOCK_DB	Fuzzy data block number
INPUT_NR	INT	Fuzzy input number

Output parameters:

Variable	Data type	Comment
INFO	BYTE	Information about execution

Meaning of the INFO parameter:

The Fuzzy_AI function informs the user about the state of the fuzzy rule base via the INFO parameter. This information is subdivided into three categories: No error, Warning and Error.

- **No error:** If execution of the function has been completed without error, the INFO parameter = '**B#16#00**'.
- **Warning:** If the START_STOP variable = 'W#16#0000' (fuzzy rule base not being processed), the INFO parameter = '**B#16#01**'. This indicates to the user that the outputs have not been updated but are actually the previous values.
- **Error:** If the programmable controller contains an instance data block with the correct length but a fuzzy rule base hasn't been loaded yet from the configuration tool, the INFO parameter = '**B#16#11**'. If the length of the specified data block is not adequate, the INFO parameter is '**B#16#21**'. If an input number (INPUT_NR) is not in the range of 1 to 8, the function sets the INFO parameter to **'B#16#31'**.

Content	Meaning
00h	No error has occurred during execution
01h	Execution of the rule base is blocked by the user or the user interface
11h	No valid fuzzy rule base is loaded in the DB (instance)
21h	Length of the data block is not adequate (no fuzzy DB)

31h	Parameter INPUT_NR not in the range of 1 to 8

Caution: If an error is found or a warning occurs the function does not enter the input values! After evaluating of the INFO parameter the user must decide whether the old values are to be processed or whether a defined value is to be outputted. If the function finds a warning or error, execution of the function is terminated immediately.

Calling the Fuzzy_AI function (FC30):

Before the fuzzy function block call, the user must enter the input values in the fuzzy data block. If you use analog inputs, the Fuzzy_AI function can be used. All parameters of this function **must** be made available by the user when it is called.

The Fuzzy_AI function normalizes an analog input signal to the upper and lower limit of the fuzzy input. The limit values are read out of the fuzzy data block and are processed as limit values in the function. The normalized value is then entered in the fuzzy data block for the relevant input.

Caution: If you use analog input signals with a bipolar range (±10 V/±20 mA...) as a fuzzy input, you must set the MAX/MIN limits of the input symmetrically (e.g. -100 and +100). If this is not the case, it can lead to false input vales being calculated in the Fuzzy_AI function.

Example call:

```
call  fc 30 (
             AI_ADR  := 348,
             BIPOLAR := TRUE,
             FUZZY_DB:= db30,
             INPUT_NR:= 1,
             INFO    := mb30
             );
```

Control of the FC:

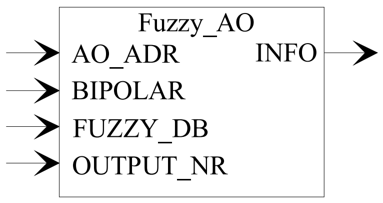
Execution of the Fuzzy_AI function **must** be controlled via the START_STOP variable in the fuzzy data block. The user can control and read this variable. The configuration tool can also change the START_STOP variable.

- The **user** can control execution of the function by the START_STOP variable. If the START_STOP variable = '**W#16#0000**', the function is **not executed**. If the START_STOP variable is not equal to '**W#16#0000**', the function is **executed**.

- The *FUZZYCONTROL++* configuration tool also controls execution of the function. Before transfer of a fuzzy system, execution is stopped with by an entry 'W#16#0000' in the START_STOP variable. After a successful transmission, the user interface enters a value not equal to '**W#16#FFFF**' in the START_STOP variable, which allows execution to resume.

6.1.14.2 FC31: Write Analog Value

The Fuzzy_AO function normalizes a fuzzy output with the upper and lower limits defined in fuzzy the rule base and writes this value to the analog output. The block must be called for each output channel of the fuzzy system that is assigned to the analog output in the user program. The user can freely assign the number of the Fuzzy_AO function (within the limits of the CPU). The default setting is FC31.



Input parameters:

Variable	Data type	Comment
AO_ADR	INT	Analog output address
BIPOLAR	BOOL	Bipolar or unipolar measurement
FUZZY_DB	BLOCK_DB	Fuzzy data block number
OUTPUT_NR	INT	Fuzzy output number

Output parameters:

Variable	Data type	Comment
INFO	BYTE	Execution information

Meaning of the INFO parameter:

The Fuzzy_AO function informs the user about the state of the fuzzy rule base via the INFO parameter. This information is subdivided into three categories: No error, Warning and Error.

- **No error:** If execution of the function has been completed without error, the INFO parameter = '**B#16#00**'.

- **Warning:** If the START_STOP variable = 'W#16#0000' (fuzzy rule base not being processed), the INFO parameter = '**B#16#01**'. This indicates to the user that the outputs have not been updated but are actually the previous values. The Fuzzy_AO function does **not** write a new analog value to the analog output.

- **Error:** If the programmable controller contains an instance data block with the correct length but a fuzzy rule base has not yet been loaded from the configuration tool, the INFO parameter = '**B#16#11**'. If the length of the specified data block is not adequate, the INFO parameter = '**B#16#21**'. If an output number (OUTPUT_NR) does not lie in the range 1 to 4, the function sets the INFO parameter to **'B#16#31'**.

Content	Meaning
00h	No error has occurred during execution
01h	Execution of the rule base is blocked by the user or the user interface
11h	No valid fuzzy rule base is loaded in the DB (instance)
21h	Length of the data block is not adequate (no fuzzy DB)
31h	Parameter OUTPUT_NR is not in the range 1 to 4

Caution: If an error is found or a warning occurs the function does not return the output values! After evaluating of the INFO parameter the user must decide whether the old values are to be processed or whether a defined value is to be outputted. If the function finds a warning or error, execution of the function is terminated immediately!

Call of the Fuzzy_AO function (FC31):

After calling the fuzzy function block, the user must read the output values out of the fuzzy data block. If you use analog outputs, the Fuzzy_AO function can be used. When this function is called all parameters **must** be made available by the user.

The Fuzzy_AO function reads a fuzzy output and normalizes it. The upper and lower limits are read out of the fuzzy data block and are processed as value limits in the function. The value normalized in this way is then written to the analog output

Note: If you use analog output signals with a bipolar range (±10 V/±20 mA...) as the fuzzy output, you must set the MAX/MIN limits of the output symmetrically (e.g. -100 and +100). If this is not the case, it can lead to false output vales being calculated in the Fuzzy_AO function.

Example call:

```
call  fc 31 (
             AI_ADR    := 348,
             BIPOLAR   := TRUE,
             FUZZY_DB  := db30,
             OUTPUT_NR:= 1,
             INFO      := mb30
             );
```

Control of the FC:

Execution of the Fuzzy_AO function **must** be controlled via the START_STOP variable in the fuzzy data block. The user can control and read this variable. The configuration tool can also change the START_STOP variable.

The **user** can control execution of the function by the START_STOP variable. If the START_STOP variable = '**W#16#0000**', the function is **not executed**. If the START_STOP variable is not equal to '**W#16#0000**', the function is **executed**.

The *FUZZYCONTROL++* configuration tool also controls execution of the function. Before transfer of a fuzzy system, execution is stopped by an entry 'W#16#0000' in the START_STOP variable. After transmission, the user interface enters the value '**W#16#FFFF**' in the START_STOP variable, which allows execution to resume.

6.2 SIMATIC-CFC-Components

6.2.1 General

The *FUZZYCONTROL++* CFC blocks can be used for process control in the same way as other SIMATIC S7 software components in the same programmable controllers – and also in conjunction with the functionality of other blocks.

The algorithms for a particular fuzzy application are calculated in the processor (CPU) of the SIMATIC S7 programmable controller and, more precisely, after an unconditional call by a user program or cyclically at time-controlled intervals. The results are stored in the associated instance DB and forwarded to the peripherals accordingly.

Each fuzzy application is represented by **two** DBs. With the CFC component, the data block is automatically determined (Instance data block = IDB). The required data is continuously exchanged between this IDB and the data block, determined in ***FUZZYCONTROL++*** (**Global data block = GDB**), so that the data is always ready to be called.

Knowledge of SIEMENS STEP 7 software is necessary for the interaction with the S7 blocks.

It is reccommended, to run FuzzyControl V5 after installing the STEP7 software package. Otherwise some elements must be inserted manually (see following chapter).

6.2.2 Library Contents

For the SIMATIC S7 series of controllers, the following function blocks (CFC components i.e. PCS7 components) exist for SIMATIC PCS7 applications for *FUZZYCONTROL++:*

FB 30:	CFC function block for S7-400	(2682 bytes)
FB 31:	CFC function block for S7-400	(2682 bytes)
DB 30:	Data block for FB 30	(4132 bytes)
DB 31:	Data block for FB 31	(20516 bytes)

For the SIMATIC S7-400 there is a 4Kbyte data block and a larger 20Kbyte data block available. For both DBs, an FB exists which is designed for the DB. It is possible to implement several fuzzy systems with one DB for each system and a common FB for all systems.

6.2.3 Inserting the Library

The data and function blocks for the CFC-4K and CFC-20K drivers are automatically inserted during the installation of *FUZZYCONTROL++* into the FuzzyControl++V5\CFC directory in the form of the library ("FuzzyCFC_Library")("FuzzyCFC" in Explorer). If the STEP7 software has been installed beforehand, then these libraries are additionally inserted into the Siemens\Step7\S7libs directory. Additionally, during installation, the library and examples are placed in the FuzzyControl++V5 target directory.

To be able to use these function blocks on the controller, you must create a project with the SIMATIC manager. Within this project the controller used is specified and an S7 program is generated. In this program, the function blocks required are copied from the "FuzzyCFC_Library".

A connection between the fuzzy configuration tool and the component is only possible if you possess a Fuzzy CFC License (optional)

Note: You can freely change the numbers of the function blocks - within the limits of the CPU. The presets are FB30 (for the "small" DB) and FB31 (for the "large" DB). They can be renamed using the STEP 7 software.

6.2.4 Functions of the „Fuzzy_FB_4K" and „Fuzzy_FB_20K" Components

The component calculates a maximum of 4 outputs from a maximum of 8 inputs using predetermined rules defined in the Fuzzy tool. You can use the projects with S7-400 CPUs and you require the additional TEST_DB (SFC24), CREAT_DB (SFC22), BLKMOV (SFC20) und ALARM_8P (SFB35) functions.

Operating Mode

If the value at the GDB "START_STOP" input is 0, **the fuzzy-tool writes the required data (Rules etc) to the global data block**, which has been defined as "DB_No" at the input. The entry to "DB_No" can be entered as hexadecimal as well as decimal. If the value changes to non-zero, the data is copied from the GDB to the instance DB and the output values are calculated by the respective rules with the aid of inputs "INPUT1" to "INPUT8."

The component can be operated in automatic mode as well as in manual mode. If the input "LIOP_SEL" is TRUE, the switch from automatic to manual or vice versa is carried via the connectable input "AUT_L". If the input is "FALSE", the switch is carried out via the OS. A switch between modes via AUT_ON_OP is allowed or not depending on the inputs "AUTOP_EN" (manual -> automatic) and "MANOP_EN" (automatic -> manual). The enable inputs are not visible by default in the CFC-editor. The current mode is shown at the QMAN_AUT output.

The automatic mode signifies, that the calculated output values can be written to the outputs "OUTPUT1" to "OUTPUT4". If the manual mode is activated, the calculated values will not be written to the outputs "OUTPUT1" to "OUTPUT4". This is the case is used for testing, where the output values can be manually set to the desired values.

6.2.5 Startup behavior

In a new fuzzy application, the FB must be entered into the empty data block first. Only when it is entered, is the data block recognized as a fuzzy data block by the configuration tool and allowed to be defined. I.e. At the beginning the component must be run at least once so that it is initialized. The component is additionally called in OB100 (automatically) and the initialization takes place in this OB during startup

6.2.6 Message Behavior

Either the previous value or a specified value will be written at the output if there is no rule for an output in the data block. This can be decided in the fuzzy configuration-tool. For each output, if no rule applies a message will be shown in SIMATIC WinCC. Furthermore a report is sent if the fuzzy application is interrupted or a if fault occurs while processing the TEST_DB, CREATE_DB or BLK_MOV system functions (see chapter 6.2.14.). A message is also generated, if the mode is switched from automatic to manual mode and vice versa. The text and type of the message can be changed by the user if desired.

6.2.7 Component Structure

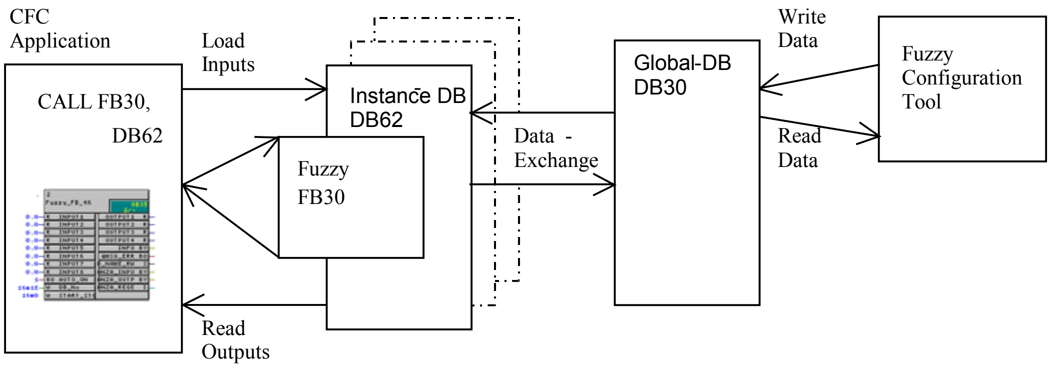
In the fuzzy **function block** (FB), all algorithms and procedures are implemented with the all the functionality that goes with high performance fuzzy application:

- Fuzzification of the inputs,

- Processing the rules,
- Defuzzification and returning the results at the outputs.

The instance **data block** (DB) in the CPU of the programmable controller and the DB, specified in *FuzzyControl++* set up the interface between the function block, the configuration tool and the user. When calling the FBs, the inputs (INPUT1,…,INPUT8) must be provided with the desired values, i.e. the inputs must be connected with the block in CFC, which returns the current values.

The outputs of the CFC function blocks can be connected to other components, to which to the output values are forwarded.

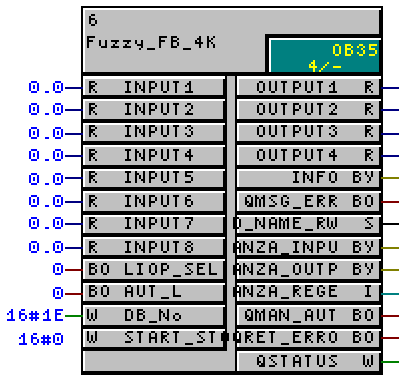


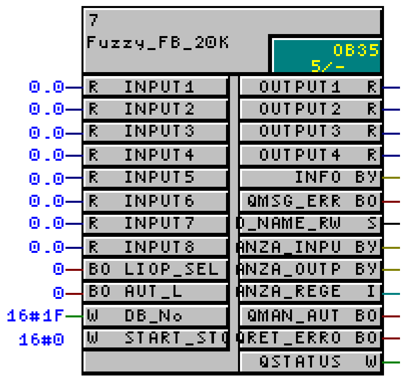
The membership functions and rules are 'implicitly' entered directly by the configuration tool into a global DB, which is specified when connecting the targetsystem. This DB number must also be entered at the input ‚DB_No' of the CFC function block. **It must be exactly the same DB-Number!** During runtime, the required, current data is exchanged between the CFC function block instance DB and the global DB.

You can transfer several fuzzy applications to- and operate them on one CPU. Every application is stored in a separate global DB and the number of the data blocks can be set freely.

6.2.8 The FB Block Diagram and Parameters

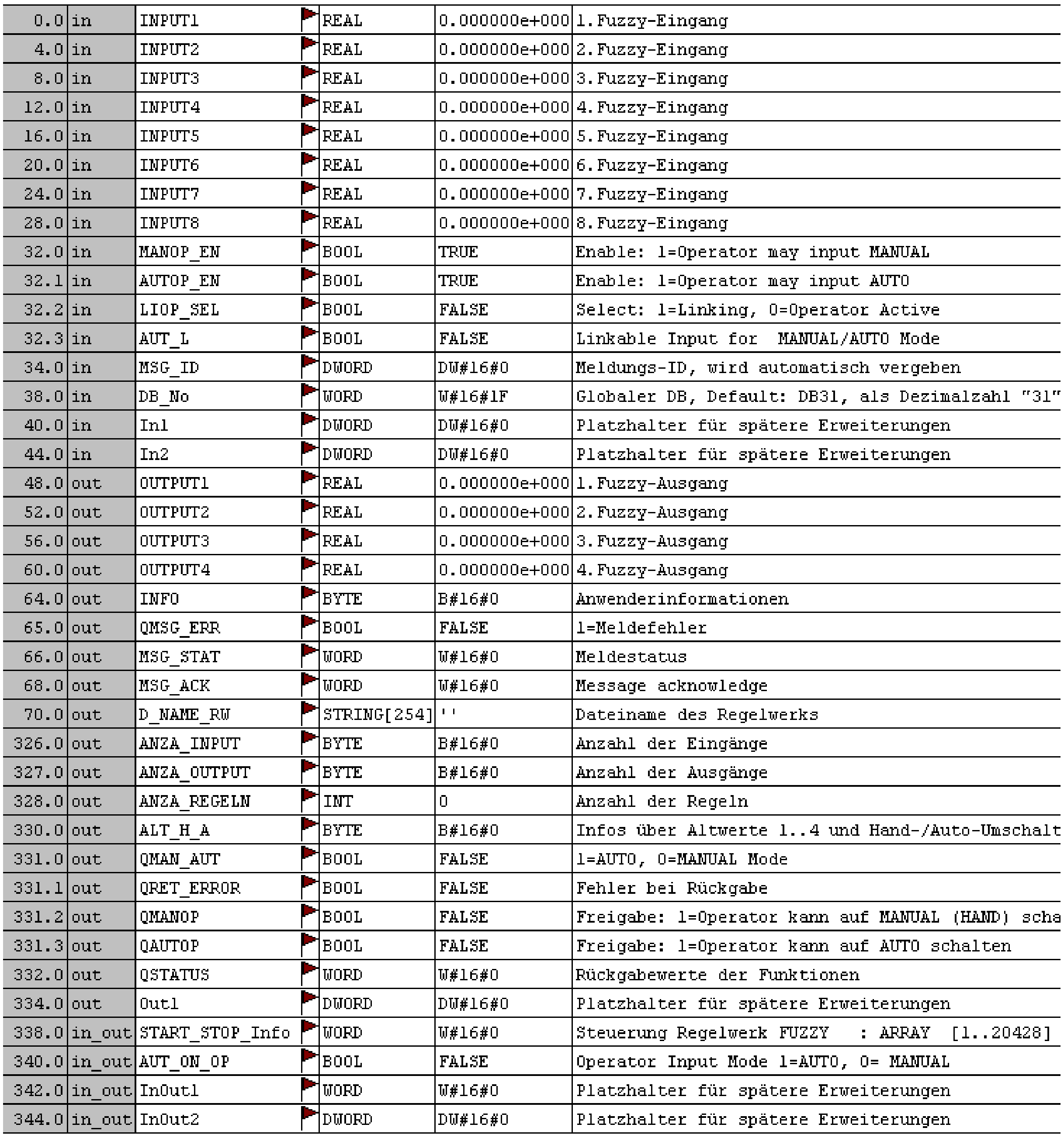
The fuzzy function blocks for CFC applications have the following Block diagrams ("Fuzzy_FB_4K"(FB30) and "Fuzzy_FB_20K"(FB31)):


6
Fuzzy_FB_4K
OB35
4/-
0.0 R INPUT1
0.0 R INPUT2
0.0 R INPUT3
0.0 R INPUT4
0.0 R INPUT5
0.0 R INPUT6
0.0 R INPUT7
0.0 R INPUT8
0 BO LIOP_SEL
0 BO AUT_L
16#1E W DB_No
16#0 W START_ST
OUTPUT1 R
OUTPUT2 R
OUTPUT3 R
OUTPUT4 R
INFO BY
QMSG_ERR BO
D_NAME_RW S
ANZA_INPU BY
ANZA_OUTP BY
ANZA_REGE I
QMAN_AUT BO
QRET_ERRO BO
QSTATUS W


7
Fuzzy_FB_20K
OB35
5/-
0.0 R INPUT1
0.0 R INPUT2
0.0 R INPUT3
0.0 R INPUT4
0.0 R INPUT5
0.0 R INPUT6
0.0 R INPUT7
0.0 R INPUT8
0 BO LIOP_SEL
0 BO AUT_L
16#1F W DB_No
16#0 W START_ST
OUTPUT1 R
OUTPUT2 R
OUTPUT3 R
OUTPUT4 R
INFO BY
QMSG_ERR BO
D_NAME_RW S
ANZA_INPU BY
ANZA_OUTP BY
ANZA_REGE I
QMAN_AUT BO
QRET_ERRO BO
QSTATUS W

Input and Output Parameters

The following table shows the data types and the structure of the FB31 input, output and additional parameters as an example.



0.0	in	INPUT1	🚩	REAL	0.000000e+000	1.Fuzzy-Eingang
4.0	in	INPUT2	🚩	REAL	0.000000e+000	2.Fuzzy-Eingang
8.0	in	INPUT3	🚩	REAL	0.000000e+000	3.Fuzzy-Eingang
12.0	in	INPUT4	🚩	REAL	0.000000e+000	4.Fuzzy-Eingang
16.0	in	INPUT5	🚩	REAL	0.000000e+000	5.Fuzzy-Eingang
20.0	in	INPUT6	🚩	REAL	0.000000e+000	6.Fuzzy-Eingang
24.0	in	INPUT7	🚩	REAL	0.000000e+000	7.Fuzzy-Eingang
28.0	in	INPUT8	🚩	REAL	0.000000e+000	8.Fuzzy-Eingang
32.0	in	MANOP_EN	🚩	BOOL	TRUE	Enable: 1=Operator may input MANUAL
32.1	in	AUTOP_EN	🚩	BOOL	TRUE	Enable: 1=Operator may input AUTO
32.2	in	LIOP_SEL	🚩	BOOL	FALSE	Select: 1=Linking, 0=Operator Active
32.3	in	AUT_L	🚩	BOOL	FALSE	Linkable Input for MANUAL/AUTO Mode
34.0	in	MSG_ID	🚩	DWORD	DW#16#0	Meldungs-ID, wird automatisch vergeben
38.0	in	DB_No	🚩	WORD	W#16#1F	Globaler DB, Default: DB31, als Dezimalzahl "31"
40.0	in	In1	🚩	DWORD	DW#16#0	Platzhalter für spätere Erweiterungen
44.0	in	In2	🚩	DWORD	DW#16#0	Platzhalter für spätere Erweiterungen
48.0	out	OUTPUT1	🚩	REAL	0.000000e+000	1.Fuzzy-Ausgang
52.0	out	OUTPUT2	🚩	REAL	0.000000e+000	2.Fuzzy-Ausgang
56.0	out	OUTPUT3	🚩	REAL	0.000000e+000	3.Fuzzy-Ausgang
60.0	out	OUTPUT4	🚩	REAL	0.000000e+000	4.Fuzzy-Ausgang
64.0	out	INFO	🚩	BYTE	B#16#0	Anwenderinformationen
65.0	out	QMSG_ERR	🚩	BOOL	FALSE	1=Meldefehler
66.0	out	MSG_STAT	🚩	WORD	W#16#0	Meldestatus
68.0	out	MSG_ACK	🚩	WORD	W#16#0	Message acknowledge
70.0	out	D_NAME_RW	🚩	STRING[254]	''	Dateiname des Regelwerks
326.0	out	ANZA_INPUT	🚩	BYTE	B#16#0	Anzahl der Eingänge
327.0	out	ANZA_OUTPUT	🚩	BYTE	B#16#0	Anzahl der Ausgänge
328.0	out	ANZA_REGELN	🚩	INT	0	Anzahl der Regeln
330.0	out	ALT_H_A	🚩	BYTE	B#16#0	Infos über Altwerte 1..4 und Hand-/Auto-Umschalt
331.0	out	QMAN_AUT	🚩	BOOL	FALSE	1=AUTO, 0=MANUAL Mode
331.1	out	QRET_ERROR	🚩	BOOL	FALSE	Fehler bei Rückgabe
331.2	out	QMANOP	🚩	BOOL	FALSE	Freigabe: 1=Operator kann auf MANUAL (HAND) scha
331.3	out	QAUTOP	🚩	BOOL	FALSE	Freigabe: 1=Operator kann auf AUTO schalten
332.0	out	QSTATUS	🚩	WORD	W#16#0	Rückgabewerte der Funktionen
334.0	out	Out1	🚩	DWORD	DW#16#0	Platzhalter für spätere Erweiterungen
338.0	in_out	START_STOP_Info	🚩	WORD	W#16#0	Steuerung Regelwerk FUZZY : ARRAY [1..20428]
340.0	in_out	AUT_ON_OP	🚩	BOOL	FALSE	Operator Input Mode 1=AUTO, 0= MANUAL
342.0	in_out	InOut1	🚩	WORD	W#16#0	Platzhalter für spätere Erweiterungen
344.0	in_out	InOut2	🚩	DWORD	DW#16#0	Platzhalter für spätere Erweiterungen

6.2.9 Calling the Function block

The function block must be called by the user. The call can be made on a cyclically and/or on a time-controlled program execution level. It is controlled via the START_STOP variable in the global DB.

When inserting the function block in CFC, an instance DB is automatically allocated and this selection cannot be changed directly. You only need to enter the required parameters from the rule base into the built-in FB. Unused inputs or outputs must not be connected. The inputs “INPUT1” to “INPUT8” as well as “OUTPUT1” to “OUTPUT4” can be connected if desired and thus “read” or as the case my be “write” the values to another block.

The connection to the configuration tool is made by specifying a global DB in the tool itself and the number of the function block at the “DB_No” input.

Following that, the program can be transferred to the controller and the controller can be switched to RUN mode. When testing the fuzzy system, the inputs can be supplied with values, e.g. via the S7 variable table or via the CFC editor, and afterwards the values calculated by the fuzzy system can be read out from the outputs. The output INFO provides information in three subdivided categories: No Error, Warning or Error (see chapter 6.2.13).Furthermore, the QRET_ERROR and QSTATUS outputs return errors during the processing of the FBs (see chapter 6.2.14)

Caution: If a block is loaded into the CPU after a change to the program, only a load of the changes is allowed to be carried out because with a complete load, all blocks on the CPU are deleted – also the new global-DBs, which contains the fuzzy data. With a complete load all data is lost and the fuzzy-system must be reloaded into the CPU.

6.2.10 External setting of the I/Os

External access to the fuzzy application in the program is possible if the corresponding variable is directly addressed in the instance DB.

Example:

```
..                          Symbolic notation of DB
..
T    "Pendulum".INPUT1
..                          Variable in DB
L    "Pendulum".OUTPUT2
..
usw
```

6.2.11 Execution control

Execution of the fuzzy application **must** be controlled via the START_STOP variable in the data block. This variable can be controlled and read by the operator. The configuration tool also changes the START_STOP variable during the data transfer.

You can directly influence execution of the fuzzy application via the START_STOP variable:

START_STOP	**Meaning**
=W#16#0000	The fuzzy application is **not** being executed
≠W#16#0000	The fuzzy application is being executed

Note: If you do not want a fuzzy application to be executed cyclically, you can control execution with the value of the START_STOP variable: e.g. by calling the function block in OB1 and by forming a timeslice in the time-controlled OB.

6.2.12 Influence using the Configuration Tool

The execution of the fuzzy application is also controlled by the configuration tool. Before transmission of fuzzy data to the DB from the PG/PC is carried out, execution is stopped by setting the START_STOP variable to W#16#0000. After transmission, the tool enters the value not equal to W#16#0000 in the START_STOP variable, which allows execution to resume.

<u>**Example:**</u>

```
STL                             Explanation
L   0
T   "Pendulum".START_STOP       Rule base is not processed

l   123
```

```
T   "Pendulum".START_STOP     Rule base is processed


I  "Pendulum".START_STOP
l   W#16#FFFF
==I
=   M 10.0                    Change of the fuzzy
                              application by the
                              configuration tool
```

Note: The configuration tool always allows execution after a transmission even if the START_STOP variable was set to W#16#0000 by the user program before transmission.

6.2.13 Evaluating the "INFO" Parameter

The block provides information about the state of the fuzzy application via the INFO parameter. This information is subdivided into three categories: No error, Warning and Error.

- **No error**

  If execution of the fuzzy application was completed without error, the variable INFO = "B#16#00".

- **Warning**

  If the START_STOP variable = W#16#0000 (fuzzy application not being executed), the INFO parameter = B#16#01.

  This shows you that the outputs have not been updated, but are actually the old values.

- **Error**
  - If an instance DB of the correct length is present in the CPU but no fuzzy application has been loaded from the configuration tool, the INFO parameter = B#16#11.
  - If the length of the instance DB specified is inadequate, the INFO parameter = B#16#21.

Content (B#16#...)	Interpretation
00	No error has occurred during execution
01	Execution of the rule base is blocked by the user or the configuration tool
11	The instance DB contains no valid fuzzy rule base
21	Length of the data block is inadequate (no fuzzy DB)

Note: If an error is found or a warning occurs, the outputs are not deleted. After evaluating the INFO variable, you must decide whether the old output values are to be processed or whether a defined value should be outputted.

Caution: If the FB finds a warning or error, execution of the function block is terminated immediately.

6.2.14 Error Informationen for QRET_ERROR and QSTATUS

If TRUE is shown at the QRET_ERROR output then an error has occurred during processing. With the aid of the QSTATUS output, it can be established, which function triggered the error and which error code has been returned.

An explanation of each error code can be found in the help section of the respective function.

Error bit	Function	Error code (W#16#...)
00000001 X0000000	TEST_DB	80A1
00000010 X0000000	TEST_DB	80B1
00000100 X0000000	TEST_DB	80B2
00001000 X0000000	TEST_DB	Parameter ‚WriteProt‘ = 1
00010000 X0000000	CREATE_DB	8091
00100000 X0000000	CREATE_DB	8092
01000000 X0000000	CREATE_DB	80A1
10000000 X0000000	CREATE_DB	80A2
00000000 X0000001	CREATE_DB	80B1
00000000 X0000010	CREATE_DB	80B2
00000000 X0000100	CREATE_DB	80B3
00000000 X0001000	BLK_MOV	8091

00000000 X0010000	BLK_MOV	8092
00000000 X0100000	BLK_MOV	8091
00000000 X1000000	BLK_MOV	8092

When an error occurs, the processing of the fuzzy block is cancelled.

6.2.15 Online Help

For each of the components FB30 and FB 31, an online-help system has been created. If you select the component and you press F1, the respective online-help appears. The corresponding entries for this are undertaken by a calling "Fuzzy_Lib.reg" in the registry. This call is carried out during the installation and does not need to be called again. From within this help file you can start the fuzzy-configuration tool. For this to work, the installation directory of the fuzzy tools must be entered into the PATH variables. This entry is atomatically made during installation of *FUZZYCONTROL++* V5.

6.2.16 Installing the Library

The "FuzzyCFC_Library" is automatically inserted into the \Siemens\Step7\S7Libs directory (if available) and the FuzzyControl++V5\CFC directory during installation. The online help file FUZZY_HELP.HLP is contained here. This file is automatically copied into the \Siemens\Step7\S7hlp directory. If the Step7 software is not installed beforehand, this file must be copied into the \Siemens\Step7\S7hlp directoryIf this file is stored in a different directory, the corresponding "Fuzzy_Lib.reg" reg-file should be adapted to suit it. If this isn't done, the online help will not work for the components

6.2.17 „Fuzzy_FB_4K" and „Fuzzy_FB_20K" Picture Components

The „Fuzzy_FB_4K" and „Fuzzy_FB_20K" blocks can be visualized in SIMATIC PCS7 V5.2 SP2 or higher with the help of 2 ready-made picture components.

During installation of *FUZZYCONTROL++* V5, the sample projects and the picture components for SIMATIC PCS7 V5 and SIMATIC PCS7 V6 are inserted in the *FUZZYCONTROL++* V5 directory. The user must use the relevant file depending on the version of SIMATIC PCS7 installed.

6.2.17.1 Inserting the picture components:

Cop the required pdl file and if desired the Fuzzy_Icon.bmp file from the \FuzzyControl++V5\CFC\Faceplates directory to your WinCC directory under GraCS.

Start the WinCC Explorer after the OS transfer and open the picture, where you would like to use the picture component.

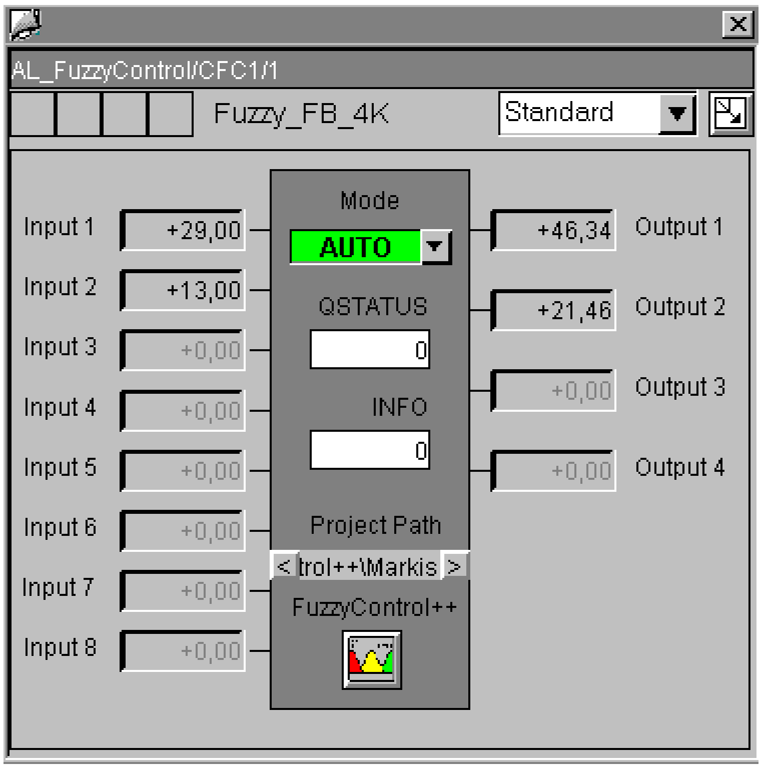
1. Insert the picture component symbol from the @Template_Fuzzy_FB.pdl file into your WinCC picture.
2. Bring the Dynamic Wizard toolbar into the editing window and open the "Standard Dynamics" tab.
3. Activate "Connect Picture Component To Measurement Point"". Acknowledge the welcoming dialog with 'Next>'.
4. Choose the relevant measuring point witht the "…" button. Click on 'Next>' and on 'Finish' in the following window.

Now the picture component is inserted in your picture and can be called after starting the runtime.

6.2.17.2 User Interfaces

- Standard-Display

With this type of display, you inputs and outputs, type of operation, errorcodes INFO and QSTATUS as well as the current project path of the fuzzy application are shown. The inputs and outputs not currently used are displayed as grayed-out. In the combo-box you can switch the operating mode from AUTO to MANUAL and vice versa.
By clicking on the arrows to the left and right of the project path, you can see the enitre path. It is possible to start *FUZZYCONTROL++* V5 with the actual fuzzy project using the button underneath 'FuzzyControl++'.


AL_FuzzyControl/CFC1/1

Fuzzy_FB_4K Standard

Input 1	+29,00	+46,34	Output 1
Input 2	+13,00	+21,46	Output 2
Input 3	+0,00	+0,00	Output 3
Input 4	+0,00	+0,00	Output 4
Input 5	+0,00		
Input 6	+0,00		
Input 7	+0,00		
Input 8	+0,00		

Mode: AUTO

QSTATUS: 0

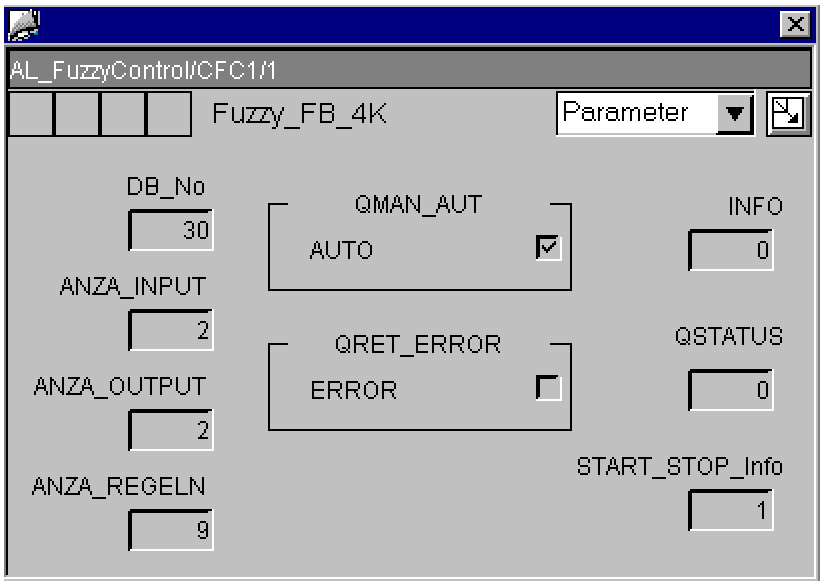
INFO: 0

Project Path: < trol++\Markis >

FuzzyControl++

- Parameter-Display

Using the selection list in the upper field ('Standard'), you can switch the display type to Parameter-Display. Here you can display the current parameters in the project relating to the fuzzy application. No changes can be made to the parameters.


AL_FuzzyControl/CFC1/1

Fuzzy_FB_4K Parameter

DB_No: 30

ANZA_INPUT: 2

ANZA_OUTPUT: 2

ANZA_REGELN: 9

QMAN_AUT: AUTO ☑

QRET_ERROR: ERROR ☐

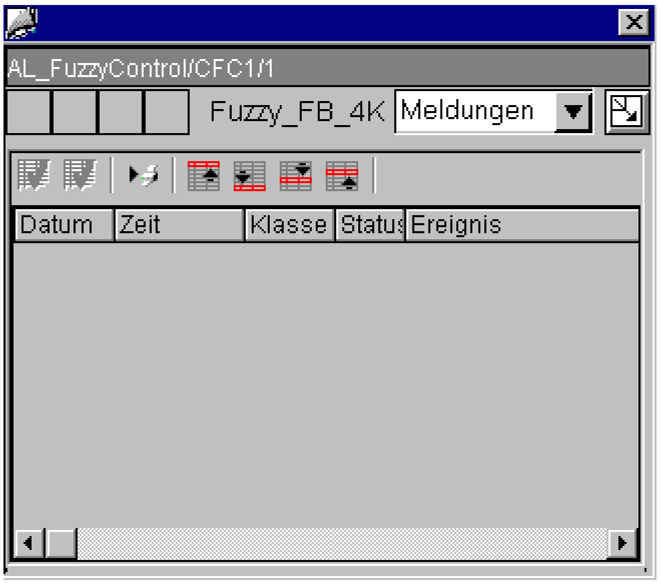
INFO: 0

QSTATUS: 0

START_STOP_Info: 1

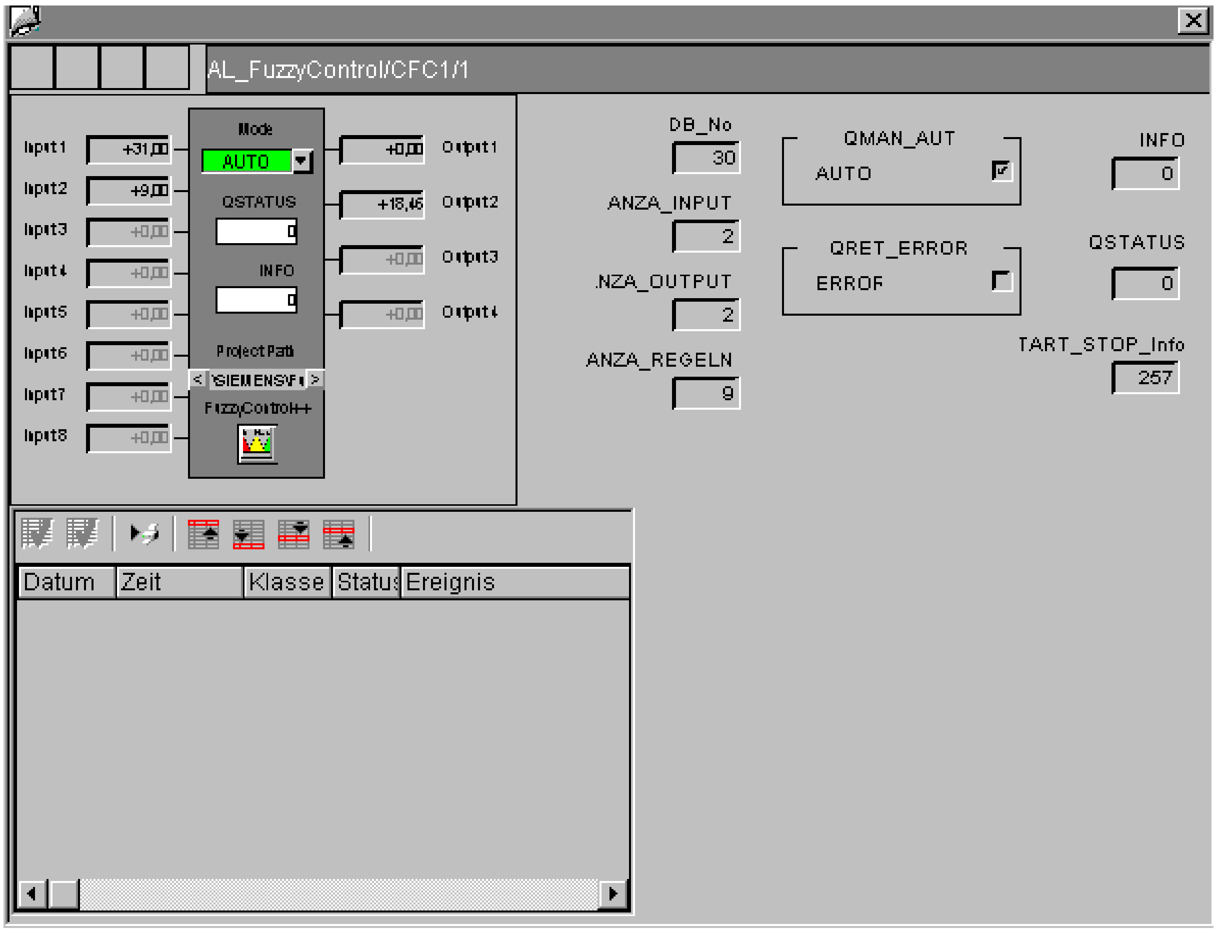
- Message-Display

Using the selection list you have the futher option of showing the Message-Display.



- Overview-Display

By clicking on the white rectangle to the right of the selection list, all 3 displays are shown in 1 picture.


AL_FuzzyControl/CFC1/1

Input1	+31,00	Output1	+0,00
Input2	+9,00	Output2	+18,46
Input3	+0,00	Output3	+0,00
Input4	+0,00	Output4	+0,00
Input5	+0,00		
Input6	+0,00		
Input7	+0,00		
Input8	+0,00		

Mode: AUTO
QSTATUS: 0
INFO: 0
Project Path: \SIEMENS\F
FuzzyControl++

DB_No: 30
ANZA_INPUT: 2
NZA_OUTPUT: 2
ANZA_REGELN: 9
QMAN_AUT: AUTO ✓
QRET_ERROR: ERROR
INFO: 0
QSTATUS: 0
TART_STOP_Info: 257

Datum	Zeit	Klasse	Statu	Ereignis

Note: If *FUZZYCONTROL++* is also to be used on an OS client, the following directions should be followed. Install *FUZZYCONTROL++* according to the installation instructions and setup the autorization on the computer. Afterwards, connect a network drive to the project server. Start *FUZZYCONTROL++* and open you user project, which is on the server, via the connected network drive.

Now you can read the fuzzy application and change it if necessary. Writing the Fuzzy Application in the DB to the AS, i.e. reading the application from the DB to the AS is, however, only possible from the server because only the server has a direct connection to the AS

6.2.18 Typical execution times

Execution of fuzzy functions are computationally intensive operations. The execution speed of specific fuzzy applications depends on the performance of the CPU used. The more often the CPU must calculate the output variables per unit time, the lower the number of fuzzy systems that can be installed

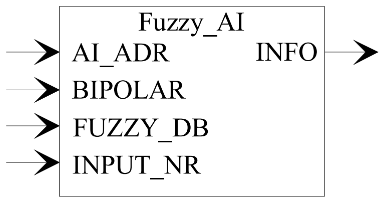
Processing times differ because of the dependence on the number of inputs/outputs, number of rules and the programmable controller. The following runtime measurements provide an indication of runtimes (S7-400 with CPU 416-2DP):

Inputs Membership Functions	2 5 each	2 5 each
Rules	10	20
Outputs Membership Functions	2 5	2 5
Runtime on S7-400 CPU 416 2DP	**Approx. 0.75 ms**	**Approx. 1.1 ms**

6.2.19 Reading (FC30) and Writing (FC31) Analog Values

6.2.19.1 FC30: Read Analog Value

The Fuzzy_AI function normalizes an analog input (voltage or current values) with the appropriate upper and lower limits, which are defined in the fuzzy rule base, and enters the normalized value into the instance data block. In the user program, the block must be called for each input channel of the fuzzy system that is set directly with an analog input. The user can freely assign the number of the Fuzzy_AI function (within the limits of the CPU). The default setting is FC30.



Input parameters:

Variable	Data type	Comment
AI_ADR	INT	Address of the analog input
BIPOLAR	BOOL	Bipolar or unipolar measurement
FUZZY_DB	BLOCK_DB	Fuzzy data block number
INPUT_NR	INT	Fuzzy input number

Output parameters:

Variable	Data type	Comment
INFO	BYTE	Information about execution

Meaning of the INFO parameter:

The Fuzzy_AI function informs the user about the state of the fuzzy rule base via the INFO parameter. This information is subdivided into three categories: No error, Warning and Error.

- **No error:** If execution of the function has been completed without error, the INFO parameter = '**B#16#00**'.

- **Warning:** If the START_STOP variable = 'W#16#0000' (fuzzy rule base not being processed), the INFO parameter = '**B#16#01**'. This indicates to the user that the outputs have not been updated but are actually the previous values.

- **Error:** If the programmable controller contains an instance data block with the correct length but a fuzzy rule base hasn't been loaded yet from the configuration tool, the INFO parameter = '**B#16#11**'. If the length of the specified data block is not adequate, the INFO parameter is '**B#16#21**'. If an input number (INPUT_NR) is not in the range of 1 to 8, the function sets the INFO parameter to **'B#16#31'**.

Content	Meaning
00h	No error has occurred during execution
01h	Execution of the rule base is blocked by the user or the user interface
11h	No valid fuzzy rule base is loaded in the DB (instance)
21h	Length of the data block is not adequate (no fuzzy DB)
31h	Parameter INPUT_NR not in the range of 1 to 8

Caution: If an error is found or a warning occurs the function does not enter the input values! After evaluating of the INFO parameter the user must decide whether the old values are to be processed or whether a defined value is to be outputted. If the function finds a warning or error, execution of the function is terminated immediately.

Calling the Fuzzy_AI function (FC30):

Before the fuzzy function block call, the user must enter the input values in the fuzzy data block. If you use analog inputs, the Fuzzy_AI function can be used. All parameters of this function **must** be made available by the user when it is called.

The Fuzzy_AI function normalizes an analog input signal to the upper and lower limit of the fuzzy input. The limit values are read out of the fuzzy data block and are processed as limit values in the function. The normalized value is then entered in the fuzzy data block for the relevant input.

Caution: If you use analog input signals with a bipolar range (±10 V/±20 mA...) as a fuzzy input, you must set the MAX/MIN limits of the input symmetrically (e.g. -100 and +100). If this is not the case, it can lead to false input vales being calculated in the Fuzzy_AI function.

Example call:

```
call  fc 30 (
             AI_ADR  := 348,
             BIPOLAR := TRUE,
             FUZZY_DB:= db30,
             INPUT_NR:= 1,
             INFO    := mb30
             );
```

Control of the FC:

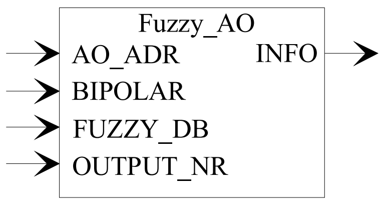
Execution of the Fuzzy_AI function **must** be controlled via the START_STOP variable in the fuzzy data block. The user can control and read this variable. The configuration tool can also change the START_STOP variable.

- The **user** can control execution of the function by the START_STOP variable. If the START_STOP variable = '**W#16#0000**', the function is **not executed**. If the START_STOP variable is not equal to '**W#16#0000**', the function is **executed**.

- The *FUZZYCONTROL*++ configuration tool also controls execution of the function. Before transfer of a fuzzy system, execution is stopped with by an entry 'W#16#0000' in the START_STOP variable. After a successful transmission, the user interface enters a value not equal to '**W#16#FFFF**' in the START_STOP variable, which allows execution to resume.

6.2.19.2 FC31: Write Analog Value

The Fuzzy_AO function normalizes a fuzzy output with the upper and lower limits defined in fuzzy the rule base and writes this value to the analog output. The block must be called for each output channel of the fuzzy system that is assigned to the analog output in the user program. The user can freely assign the number of the Fuzzy_AO function (within the limits of the CPU). The default setting is FC31.



Input parameters:

Variable	Data type	Comment
AO_ADR	INT	Analog output address
BIPOLAR	BOOL	Bipolar or unipolar measurement
FUZZY_DB	BLOCK_DB	Fuzzy data block number
OUTPUT_NR	INT	Fuzzy output number

Output parameters:

Variable	Data type	Comment
INFO	BYTE	Execution information

Meaning of the INFO parameter:

The Fuzzy_AO function informs the user about the state of the fuzzy rule base via the INFO parameter. This information is subdivided into three categories: No error, Warning and Error.

- **No error:** If execution of the function has been completed without error, the INFO parameter = '**B#16#00**'.
- **Warning:** If the START_STOP variable = 'W#16#0000' (fuzzy rule base not being processed), the INFO parameter = '**B#16#01**'. This indicates to the user that the outputs have not been updated but are actually the previous values. The Fuzzy_AO function does **not** write a new analog value to the analog output.
- **Error:** If the programmable controller contains an instance data block with the correct length but a fuzzy rule base has not yet been loaded from the configuration tool, the INFO parameter = '**B#16#11**'. If the length of the specified data block is not adequate, the INFO parameter = '**B#16#21**'. If an output number (OUTPUT_NR) does not lie in the range 1 to 4, the function sets the INFO parameter to **'B#16#31'**.

Content	Meaning
00h	No error has occurred during execution
01h	Execution of the rule base is blocked by the user or the user interface
11h	No valid fuzzy rule base is loaded in the DB (instance)
21h	Length of the data block is not adequate (no fuzzy DB)
31h	Parameter OUTPUT_NR is not in the range 1 to 4

Caution: If an error is found or a warning occurs the function does not return the output values! After evaluating of the INFO parameter the user must decide whether the old values are to be processed or whether a defined value is to be outputted. If the function finds a warning or error, execution of the function is terminated immediately!

Call of the Fuzzy_AO function (FC31):

After calling the fuzzy function block, the user must read the output values out of the fuzzy data block. If you use analog outputs, the Fuzzy_AO function can be used. When this function is called all parameters **must** be made available by the user.

The Fuzzy_AO function reads a fuzzy output and normalizes it. The upper and lower limits are read out of the fuzzy data block and are processed as value limits in the function. The value normalized in this way is then written to the analog output

Caution: If you use analog output signals with a bipolar range (±10 V/±20 mA...) as the fuzzy output, you must set the MAX/MIN limits of the output symmetrically (e.g. -100 and +100). If this is not the case, it can lead to false output vales being calculated in the Fuzzy_AO function.

Example call:

```
call  fc 31 (
             AI_ADR    := 348,
             BIPOLAR   := TRUE,
             FUZZY_DB  := db30,
             OUTPUT_NR:= 1,
             INFO      := mb30
             );
```

Control of the FC:

Execution of the Fuzzy_AO function **must** be controlled via the START_STOP variable in the fuzzy data block. The user can control and read this variable. The configuration tool can also change the START_STOP variable.

The **user** can control execution of the function by the START_STOP variable. If the START_STOP variable = '**W#16#0000**', the function is **not executed**. If the START_STOP variable is not equal to '**W#16#0000**', the function is **executed**.

The *FUZZYCONTROL*++ configuration tool also controls execution of the function. Before transfer of a fuzzy system, execution is stopped by an entry ‘W#16#0000’ in the START_STOP variable. After transmission, the user interface enters the value '**W#16#FFFF**' in the START_STOP variable, which allows execution to resume.

6.3 SIMATIC WinCC

6.3.1 Installing the WinCC-OLL

The corresponding OLLs for the various versions of SIMATIC WinCC can be found in the *FUZZYCONTROL++* installation directory.

During the installation of *FUZZYCONTROL++* the correct OLL will have been placed in the WinCC directory, depending on which verision of SIMATIC WinCC you have on your computer.

The following files will be copied into the WinCC bin directory (normally C:/Siemens/WinCC/bin):

• fuzzy.oll	DLL object that implements the fuzzy block in WinCC 3.1 or 4.0.
• fuzzyDLL.dll	Provides functions required by Fuzzy.oll (can also be located in the Windows system directory!)
• fuzzyenu.lng	Required for the English WinCC user interface.
• fuzzyFRA.lng	Required for the French WinCC user interface.

6.3.2 WinCC Applications with the WinCC-OLL from FuzzyControl

6.3.2.1 Editing Actions for a Fuzzy object

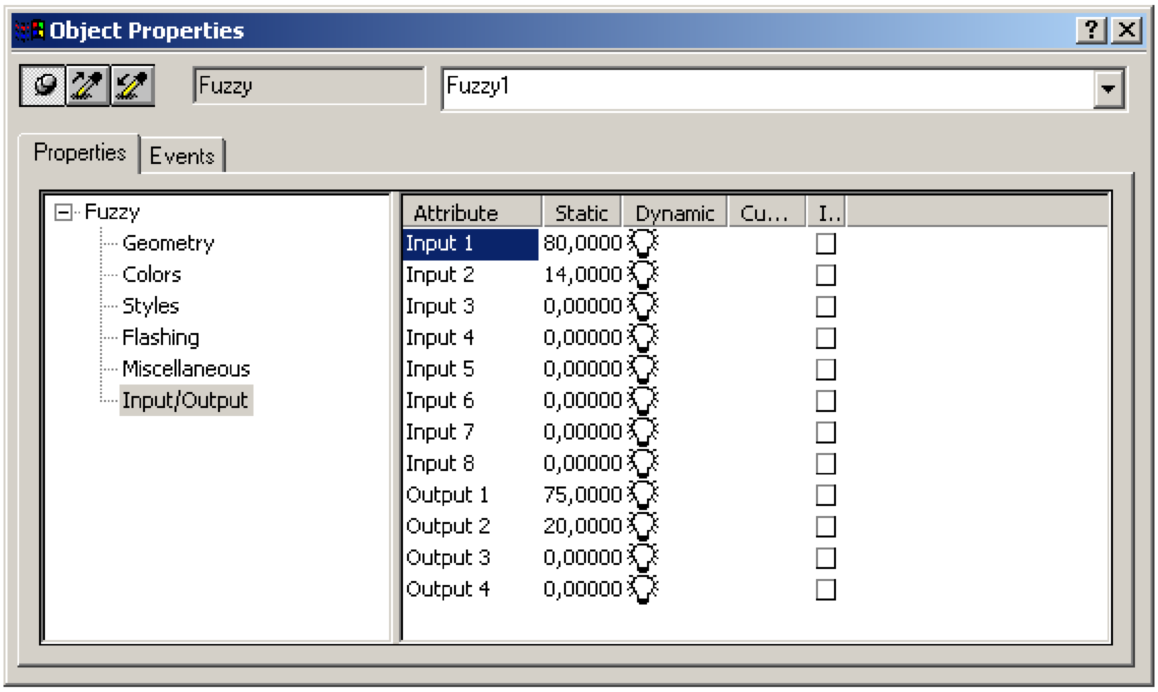
The following actions must be performed.

- Put the Fuzzy.oll into the WinnCC picture
- Assign the fpl file with the file-path
- Create internal WinCC variables
- Link the variables with the I/O field

Next, you have to edit a C-action, which

- Gets the internal variables
- Sets the fuzzy inputs
- Queries the fuzzy outputs
- Sets the internal variables as the output value

It’s also acceptable to put the C-action after input1. This C-action is then called cyclically.



Attribute	Static	Dynamic	Cu...	I..
Input 1	80,0000	💡		☐
Input 2	14,0000	💡		☐
Input 3	0,00000	💡		☐
Input 4	0,00000	💡		☐
Input 5	0,00000	💡		☐
Input 6	0,00000	💡		☐
Input 7	0,00000	💡		☐
Input 8	0,00000	💡		☐
Output 1	75,0000	💡		☐
Output 2	20,0000	💡		☐
Output 3	0,00000	💡		☐
Output 4	0,00000	💡		☐

6.3.2.2 *C-action example for fuzzy object "fuzzy1"*

```
#include "apdefap.h"
 double _main(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName)
{
double dTarget;                    // local variable
double dActual;                    // local variable
double dDiff;                      // local variable
double dFuzzyOutput;               // local variable

dtarget = GetTagDouble("target value");   // get value of the internal variable target value
dactual = GetTagDouble("actual value");    // get value of the internal variable actual value
dDiff=dTarget-dActual;                     //difference creation
         picture name    object name    1stinput

SetPropDouble("fuzzydemo","Fuzzy1","FuzzyIn1",dDiff);                  //set Fuzzy-input
                 Read output           1st output
```

```
dFuzzyOutput = GetPropDouble("fuzzydemo","Fuzzy1","FuzzyOut1");   //query Fuzzy-output
dActual = GetTagDouble("actual value");                             //query actual value

dActual = dActual - dFuzzyOutput;                                   //calculate new actual value
```

New value of internal variable.

```
SetTagDouble("actual value",dActual);                               //set actual value

return dDiff;
}
```

6.3.2.3 Simple fuzzy example

```
#include "apdefap.h"
 double _main(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName)
{
double Input-value1;
double Input-value2;
double Qutput-value1;

Input-value1 = GetTagDouble("e1");                         // get internal variable i1
Input-value2 = GetTagDouble("e2");                         // get internal variable i2

SetPropDouble("NewPdl0","Fuzzy1","FuzzyIn1",Input-value1);          //set Fuzzy-input 1
SetPropDouble("NewPdl0","Fuzzy1","FuzzyIn2",Input-value2);          //set Fuzzy-input 2

Output-value1 = GetPropDouble("NewPdl0","Fuzzy1","FuzzyOut1");   //query Fuzzy-output1
SetTagDouble("o1",Output-value1);                                  //set internal variable o1
return Output-value1;
}
```

6.3.2.4 Naming of Fuzzy In/Outputs

Fuzzy:	**Inputs:**	**Outputs:**
	FuzzyIn1	FuzzyOut1
	FuzzyIn2	FuzzyOut2
	FuzzyIn3	FuzzyOut3
	FuzzyIn4	FuzzyOut4
	FuzzyIn5	
	FuzzyIn6	
	FuzzyIn7	
	FuzzyIn8	

6.3.2.5 Limitations

Please be aware, that the WinCC-OLL fuzzy rule base is only active when the picture is active.

If you would like the fuzzy rule base to be always active, you must additionally put the WinCC-OLL into the overview picture.

Online-monitoring such as that with the SIMATIC S7 and SIMATIC CFC components is **not** possible with the WinCC-OLL.

PART III:

FUZZYCONTROL++ -

REFERENCE

This part of the manual provides a systematic overview to the possibilities of *FUZZYCONTROL++*. It is arranged according to the menu options and operating structures available in the working window.

7 Operating Structure Of *FUZZYCONTROL++*

You create a project with the *FUZZYCONTROL++* program. A fuzzy system *project* is defined by choosing its inputs and outputs (total number and properties), its targetsystem and its structure. Editing is performed in the window titled *FUZZYCONTROL++*. It displays the working window (basic window) of *FUZZYCONTROL++*.

7.1 Working Window

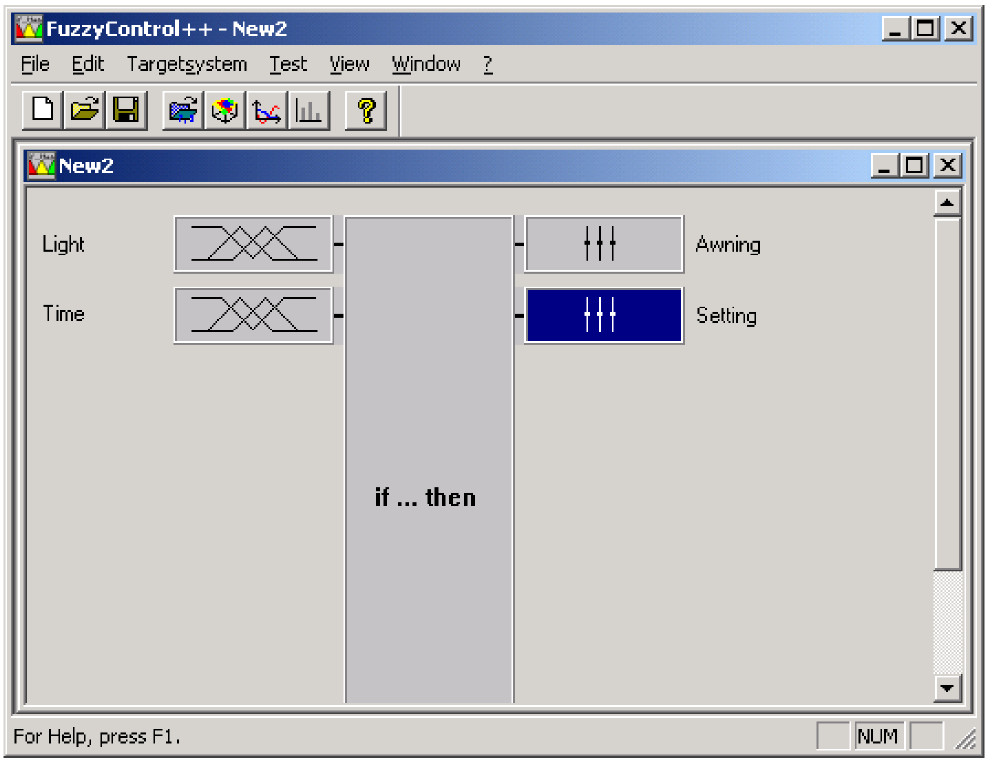
When you first start *FUZZYCONTROL++* an incomplete working window is opened containing only the *File, View* and *Help* menu commands. It does not contain a project yet. The complete working window is only displayed when you start editing a project. You can do this in one of two ways:

- By loading an existing project with *Open*
- By creating a new project with *New* and entering the external architecture (number of inputs, outputs and the targetsystem)

The complete working window (which we shall refer to as the "working window") shows the complete menu bar and a window with the block diagram of the fuzzy system. This block diagram window plays a central role in processing the project and is therefore called the project window.

All actions for editing the fuzzy system are available in the project window of the current project. They are accessed via the block symbol buttons for the in/outputs and the rule block. The actions are subdivided into two categories:

- Editing the in/outputs
- Editing the rule base



Operation of the program when you are editing a project is mainly based on the menu bar and the project window. It must remain open the whole time you are working on the project, but it can, of course, be minimized to icon size. If other windows are opened as you work on the project, they have the project name and are also given a number. The active project window is always identified by the number 1, e.g.:

"C:\.….SIEMENS\FuzzyControl++V5\Samples\ FuzzyControl++\awning.fpl:1".

If you close the project window you also close the windows associated with it and close the project.

In one session with *FUZZYCONTROL++* you can work on **more than one project** in parallel. In this case, several project windows are open in the working window (distinguished by the project name).

The menus of the working window and its sub-items are usually activated as follows

- with a click of the left mouse button on the menu name in the menu bar/menu table or
- with the <Alt + ... (letter underlined in the name in the menu)> key combination.

Other operations are listed along with the menu options under "Shortcuts:".

Example: You can close a file by:

- selecting the *Close* command in the *File* menu or with the

- <ALT+F, C> key sequence

Note: From now on, *LMB* is used for the "left mouse button" and *RMB* is used for the "right mouse button".

Menu options are activated by a single click on them with *LMB*.

7.2 Menu: File

The menu contains the commands to

- Begin and exit project work, and to
- Save copies of a project intermittently

7.2.1 New

Shortcut: On the toolbar with LMB on the icon

or <CTRL+N> key combination or <ALT+F, N> key sequence

With this command you create a new project and it opens the *Define Project* dialog box

***Define Project* window**

In the dialog box *Define Project* you can set the number of inputs and outputs and the targetsystem for the project.

To be able to start a new fuzzy project, you must define the external architecture of the fuzzy system by the number of inputs and outputs. You can define up to 8 inputs and 4 outputs for your project. The definition made here can still be modified later on in the *Edit* menu, where you can add or delete inputs and outputs.

To begin the project you must define the targetsystem-driver. The drivers for the SIMATIC S7-300, SIMATIC S7-400 and SIMATIC WinCC systems can be selected. No targetsystem is set by default. The following values apply for each targetsystem.

Targetsystem	Maximum number	Maximum number	Maximum number

	of inputs	**of outputs**	**of rules**
SIMATIC S7-300: S7-4K	8	4	200
SIMATIC S7-400: CFC-4K:			
SIMATIC S7-400 S7-20K, CFC-20K:	8	4	2000
SIMATIC WinCC:	8	4	2000

Note: The components for the S7-300 can also be loaded on a S7-400 and have to be addressed with the corresponding "little" targetsystem-drivers.

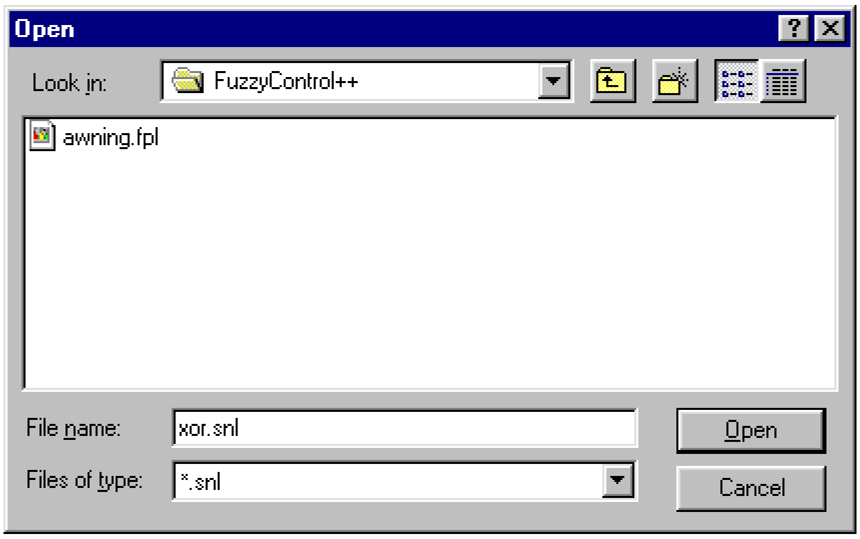
After you have closed the window by clicking *OK* the project is defined and is given the initial name *New1*. The corresponding project window is displayed with a symbolic representation of the fuzzy system. If you create several new projects one after the other, they appear with the initial names *New1, New2*, etc. in the working window *FUZZYCONTROL++*.

7.2.2 Open

Shortcut: On the toolbar with LMB on the 📂 icon

or <CTRL+O> key combination or <ALT+F, O> key sequence

Use this command if you want to resume work on an existing *FUZZYCONTROL++* project that has been saved. The Windows *Open* dialog box appears.



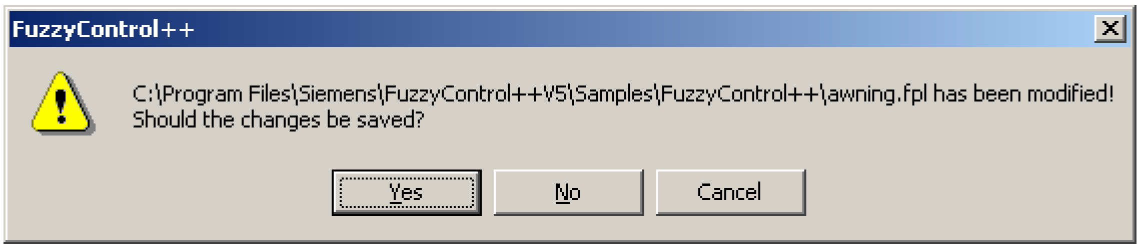
If you select the filename from the list with the LMB and click on *Open* with the LMB, the project, e.g. *awning.fpl*, is opened in the working window in the associated project window form (see figure in "7 Operating Structure of *FUZZYCONTROL++*"). With *FUZZYCONTROL++* you can edit more than one project in parallel. You can therefore open more than one project beside each other, in which case each project appears in its own project window with the project name as the title in the *FUZZYCONTROL++* working window.

7.2.3 Close

Shortcut:	In the project window	with a single click on the icon ☒
		with a double-click on the icon
or	<ALT+F, C> key sequence	

This command closes the current project and closes the project window as well as all associated windows. The *FUZZYCONTROL++* working window remains.

If changes were made to the project (which is usually the case), *FUZZYCONTROL++* asks whether you want to save the project.



C:\Program Files\Siemens\FuzzyControl++V5\Samples\FuzzyControl++\awning.fpl has been modified!
Should the changes be saved?

If you answer *Yes*, the previous version is overwritten with the current version. If you answer *No* the new changes are lost and the project remains in its previous state.

If you close a newly created project, *FUZZYCONTROL++* opens the *Save as...* dialog box, where you can define the name of the project before it is saved.

7.2.4 Save

Shortcut: On the toolbar with LMB on the 💾 icon

or <CTRL+S> key combination or <ALT+F, S> key sequence

Use this command to save the current project under its current name and directory. The project is saved with the **.fpl (FUZZY PROGRAMMING LANGUAGE)* extension.

7.2.5 Save as...

Shortcut: <ALT+F, A> key sequence

If you want to change the name or the directory of an existing project, you must choose the *Save as...* command. When a project is first saved, *FUZZYCONTROL++* displays the *Save as...* dialog box, so that you can give your project a name. A project is saved with the **.fpl (FUZZY PROGRAMMING LANGUAGE)* or **.fcl (FUZZY CONTROL LANGUAGE)* extension.

7.2.6 Commands 1, 2, 3, 4

Shortcut: <ALT+F, 1...4> key sequence

With this command you can reopen one of the last four projects that was closed. The project at the top of the list is the last one closed.

7.2.7 Close

Shortcut: In the working window of *FUZZYCONTROL++* with a single click on the ☒ icon or a double click on

or <ALT+F4> Key combination or <ALT+F, X> key sequence

Use this command to exit your *FUZZYCONTROL++* session. If changes were made to the project, *FUZZYCONTROL++* asks whether you want to save the changes.

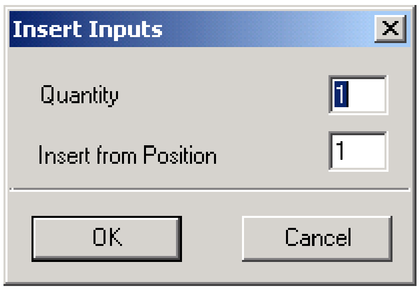
7.3 Menu: Edit

The *Edit* menu is used to add or remove inputs and outputs in the current project. It is also used to open the rule base in the rule table form.

7.3.1 Inserting inputs/outputs

Shortcut: Click on the input/output concerned with RMB in the project window and select *Insert*.

or <ALT+E, I> (input) or <ALT+E, S> (output) key sequence


Insert Inputs

Quantity: 1

Insert from Position: 1

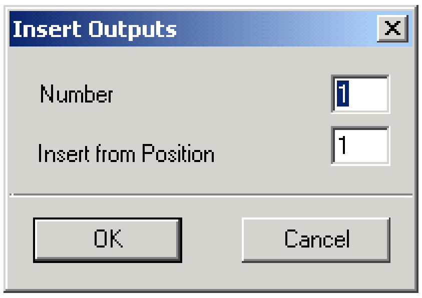
This dialog box appears for inserting inputs. You must enter the number of the inputs and the position at which you want to insert the new inputs. The topmost input in the block diagram is position 1. After confirmation with *OK*, an input is inserted at the required position and the previous inputs are shifted downwards.

Insertion of outputs is performed in the same way.

7.3.2 Deleting inputs/outputs

Shortcut: Click on the input/output concerned with RMB in the project window and select *Delete*.

or Key sequence <ALT+E, D> (input) or <ALT+E, L> (output)



You can remove inputs with this dialog box. You must state the number of the inputs you wish to delete, and the position of the first input to be deleted. After you have clicked *OK* and confirmed the action, the input at that position is deleted. The topmost input in the block diagram is in position 1. There must always be at least one input and output.

Deletion of outputs is performed in the same way.

7.3.3 Rule Table...

Shortcut: <ALT+E, R> key sequence

This menu option opens the window with the rule table of the opened project. If more than one project is opened, the rule base must be opened for the project which is active.

7.3.4 Editing inputs/outputs

You can edit the following properties for both inputs and outputs:

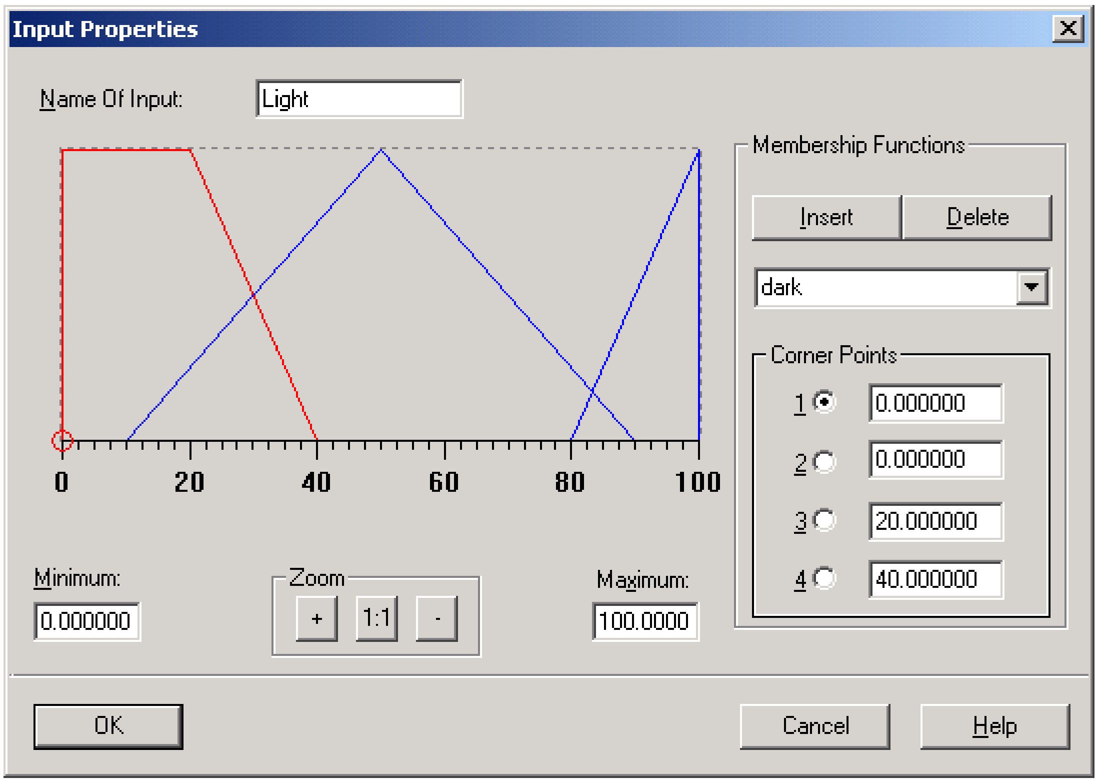
- *Name* of the input and output
- *Normalization* with respect to Minimum and Maximum of the value range
- Changing the number of membership functions by inserting and deleting them
- Renaming a membership function

We shall explain the above properties using the example of the *Input Properties* dialog box shown in the following figure.

7.3.4.1 Input Properties

Shortcut: Double-click with the LMB on the button of the input in question

or Click with the RMB on the input concerned and select *Input Properties*


Input Properties

Name Of Input: Light

Membership Functions

Insert Delete

dark

Corner Points

1 ● 0.000000

2 ○ 0.000000

3 ○ 20.000000

4 ○ 40.000000

Minimum: 0.000000

Zoom + 1:1 -

Maximum: 100.0000

OK Cancel Help

Name

In the entry field you can enter a name for this input which is relevant to the problem being solved. You can enter up to ten characters. The first character may not be a number. Special characters and accented characters are also not permitted.

Normalize

You can normalize the input by entering values in the *Maximum* and *Minimum* fields.

- Normalization without previously defined membership functions

  After you have entered the max/min values, the value range of the graphical display for the membership functions in the dialog box is adjusted to the limits you have set.

- Normalization with previously defined membership functions

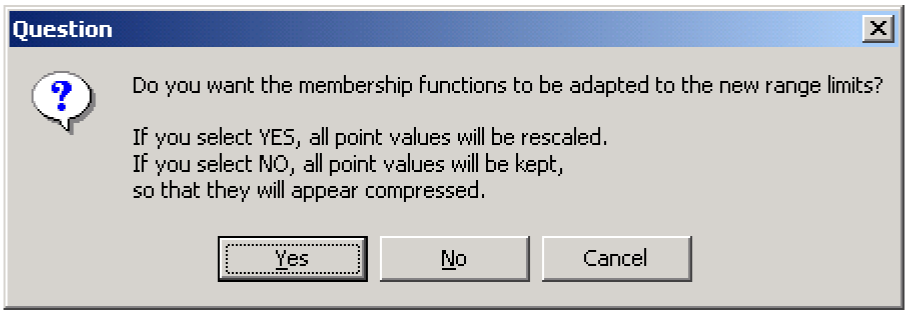
  A distinction is made between two different cases when you change the limits of previously defined membership functions:

1. The new limit values are within the range that is determined by the "external" membership functions.

   After you have entered the limits, a message box appears with the indication that the membership functions will be adapted to the new limits. They are adapted after you have confirmed with *OK*. If you dismiss the message box with *Cancel*, the fields retain the old values.

2. The new limit values are outside the range that is determined by the "external" membership functions.

   After you have entered the limits the following message box is displayed.



Do you want the membership functions to be adapted to the new range limits?

If you select YES, all point values will be rescaled.
If you select NO, all point values will be kept,
so that they will appear compressed.

Zoom

+ The range of the graphical display of the membership functions is halved.

1:1 The range of the graphical display of the membership function is reset to its original value.

- The range of the graphical display of the membership function is doubled.

Membership function

In this range of the dialog box, you can insert or delete membership functions. You can also change the names and point values of the membership functions.

- Inserting membership functions

  If membership functions are inserted for an input for the first time, a dialog box appears in which you can enter the number of membership functions to be inserted. The default value entered is 5.

  If you have already defined membership functions, only one more membership function is inserted when you click the *Insert* button.

  Note: The maximum number of membership functions is seven.

- Deleting membership functions

  To delete a membership function you must first select it from the list box below the two buttons *Insert* and *Delete*. After you have clicked the *Delete* button, this membership function is deleted.

  If all membership functions of an input are deleted, the input is irrelevant for calculating the rules.

- Renaming membership functions

  To rename a membership function you must first select it from the list box below the two buttons *Insert* and *Delete*. After that, you can change the name by entering a string (7 characters max.) in the text box below the buttons.

- Changing the point values of membership functions

You can change the value of each point of the selected membership function in one of two ways. You can move the values within the graphical display by moving individual points of the selected membership function. You can also change the point values by entering values in the point fields of the *membership function* area of the dialog box.

Please note: The following applies to the values of the points $P_1 \le P_2 \le P_3 \le P_4$

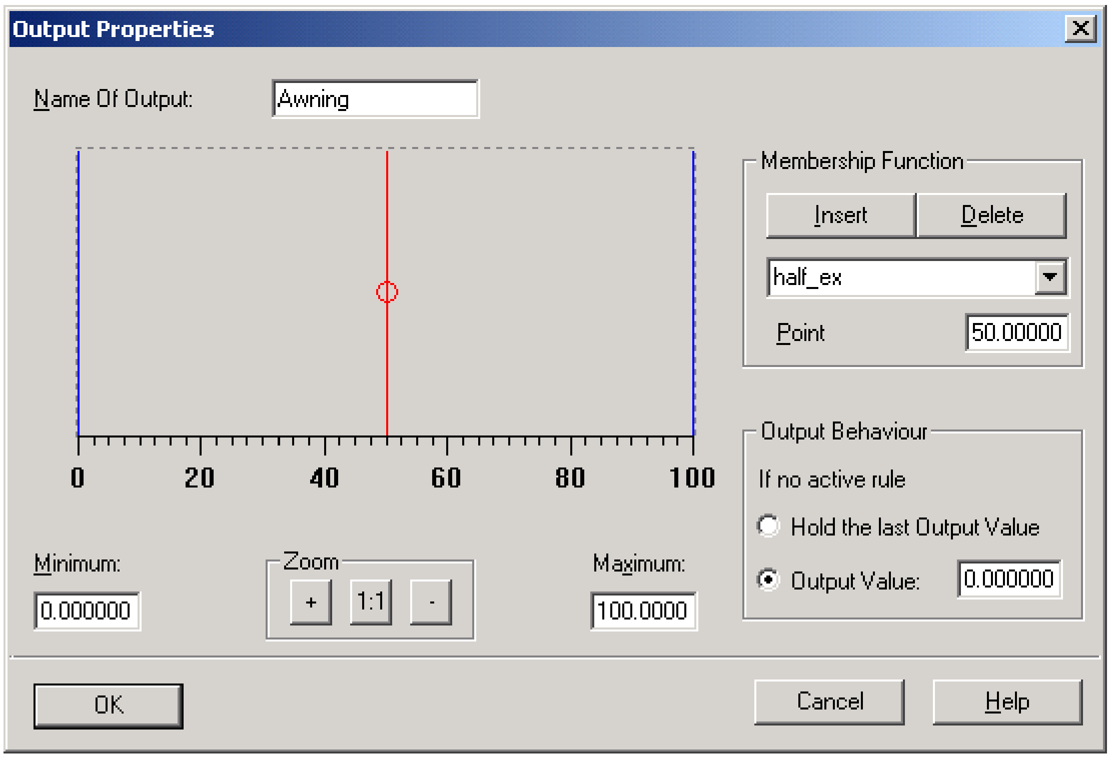
- Representation of membership functions

The membership functions for the inputs can be represented both as a triangle and as a trapezoid. If a new membership function is inserted, it is represented in the form of a triangle.

Output Properties

Shortcut: Double-click with the LMB on the button of the output in question

or Click with the RMB on the output concerned and select *Output Properties*


Output Properties

Name Of Output: Awning

Membership Function: half_ex

Point: 50.00000

Output Behaviour

If no active rule

( ) Hold the last Output Value

(●) Output Value: 0.000000

Minimum: 0.000000

Maximum: 100.0000

The appearance and handling of this dialog box is very similar to that of the *Input Properties* dialog box. This section will therefore only deal with the differences between the two dialog boxes.

Membership functions

- Representation of membership functions

  The membership functions for the outputs are represented as singletonss. Singletons are triangular membership functions pushed together to form a line. Their position on the horizontal axis is only determined by a single point value.

  Note: The maximum number of membership functions is 9.

Output behavior

In this part of the dialog box you can define the output behavior if no rule is active.

- Keep the last Output Value

  If you select this option, the value from when a rule was last active, is held.

- Predefined Output Value:

  If you select this option, the output is set to a constant value (entered in the field) if no rule is active.

7.3.5 Editing the rules

The *FUZZYCONTROL++* tool offers the following two ways of editing the rules of the fuzzy system:

- Editing rules in a rule table
- Editing rules in a rule matrix

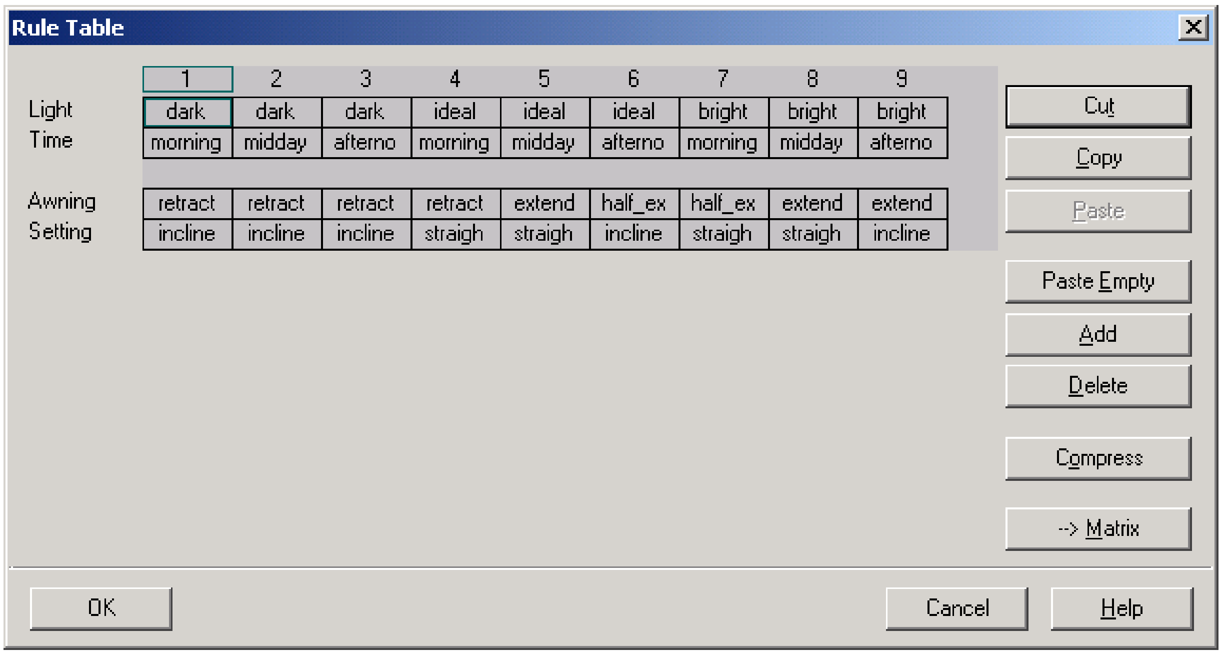
Please note: It is always possible to convert a rule matrix to a rule table, but it is not always possible to convert a rule table to a rule matrix.

7.3.5.1 Rule table

Shortcut: Double-click with the LMB on *if...then* box

or Click with the RMB on *if...then* box and select R*ule Table*

or Select menu *Edit/Rules...*



	1	2	3	4	5	6	7	8	9
Light	dark	dark	dark	ideal	ideal	ideal	bright	bright	bright
Time	morning	midday	afterno	morning	midday	afterno	morning	midday	afterno
Awning	retract	retract	retract	retract	extend	half_ex	half_ex	extend	extend
Setting	incline	incline	incline	straigh	straigh	incline	straigh	straigh	incline

The *Rule Table* shown in the window above allows you to perform the following operations:

- Define the rules for the inputs and outputs by selecting the name of the associated membership function.
- Cut out an existing rule (the rule is copied to the Windows clipboard and removed from the rule base).
- Copy an existing rule (the rule is copied to the Windows clipboard)
- Insert a rule (inserts the rule stored in the Windows clipboard at the marked position).
- Insert a new rule (inserts an empty column at the selected position in the rule base).
- Add a new rule (appends an empty column to the end of the rule base)
- Delete a rule (deletes the selected rule from the rule base)
- Compress (removes empty columns from the rule table.)
- Matrix (switches from rule table to rule matrix representation).

If the rule base is fully defined, you can close the window by clicking on *OK* .

Of course you can modify, delete or insert new rules afterwards.

Hinweis: Inverse rules (shown as „/membership functions“) can also be chosen in the rule table, but inverse rules are not possible in the rule matrix. Consequently, conversion from a rule table to a rule matrix is not possible.

7.3.5.2 Rule matrix

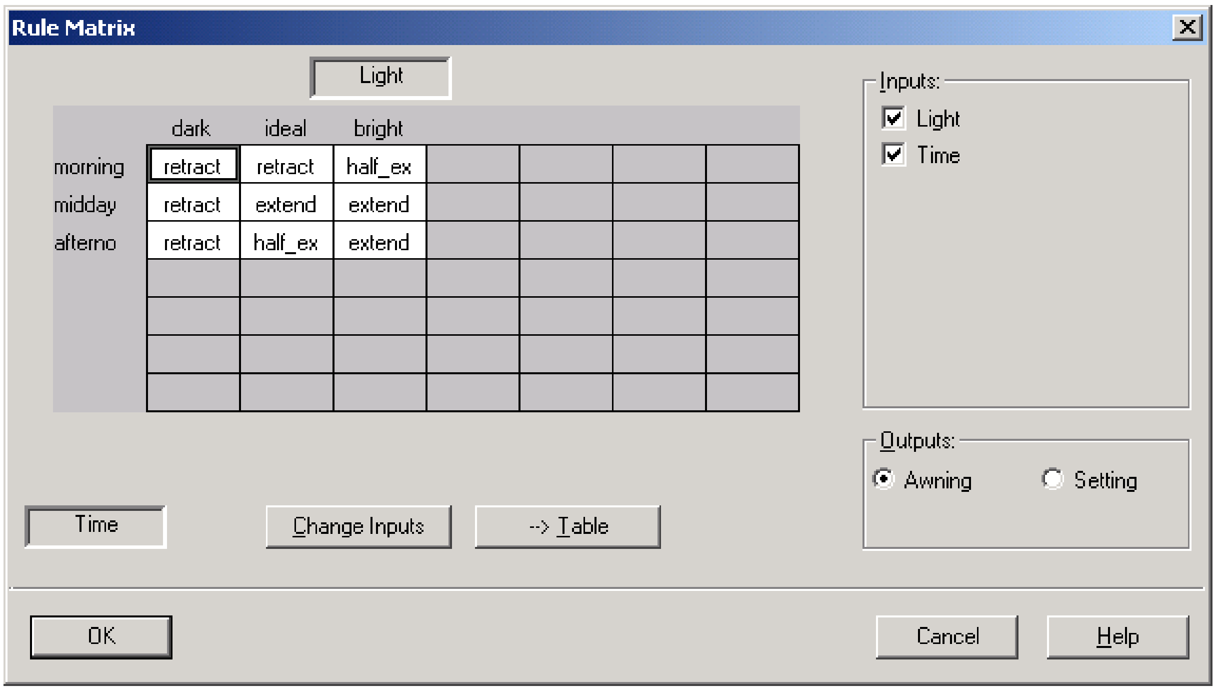
Shortcut: Click with RMB on *if...then* box and select *Rule matrix*

The R*ule Matrix* allows you to create the rule base in the form of a matrix consisting of several sheets. The names of the membership functions for the inputs are entered on the edges of the matrix. Within the matrix, for each combination of inputs, you can enter the name of the relevant membership function for the output selected.

If there are more than two inputs or more than one output, the matrix can be created for each of the additional inputs and outputs, i.e. a corresponding number of "sheets" for the rule matrix is created.

You have the following options for operation:

- Change Inputs (The assignment of inputs to the rows and columns of the matrix is swapped round.)
- Table (Switches to tabular representation of the rule base)


Rule Matrix

Light

	dark	ideal	bright				
morning	retract	retract	half_ex				
midday	retract	extend	extend				
afterno	retract	half_ex	extend				

Inputs: [✓ Light] [✓ Time]

Outputs: [✓ Awning] Setting

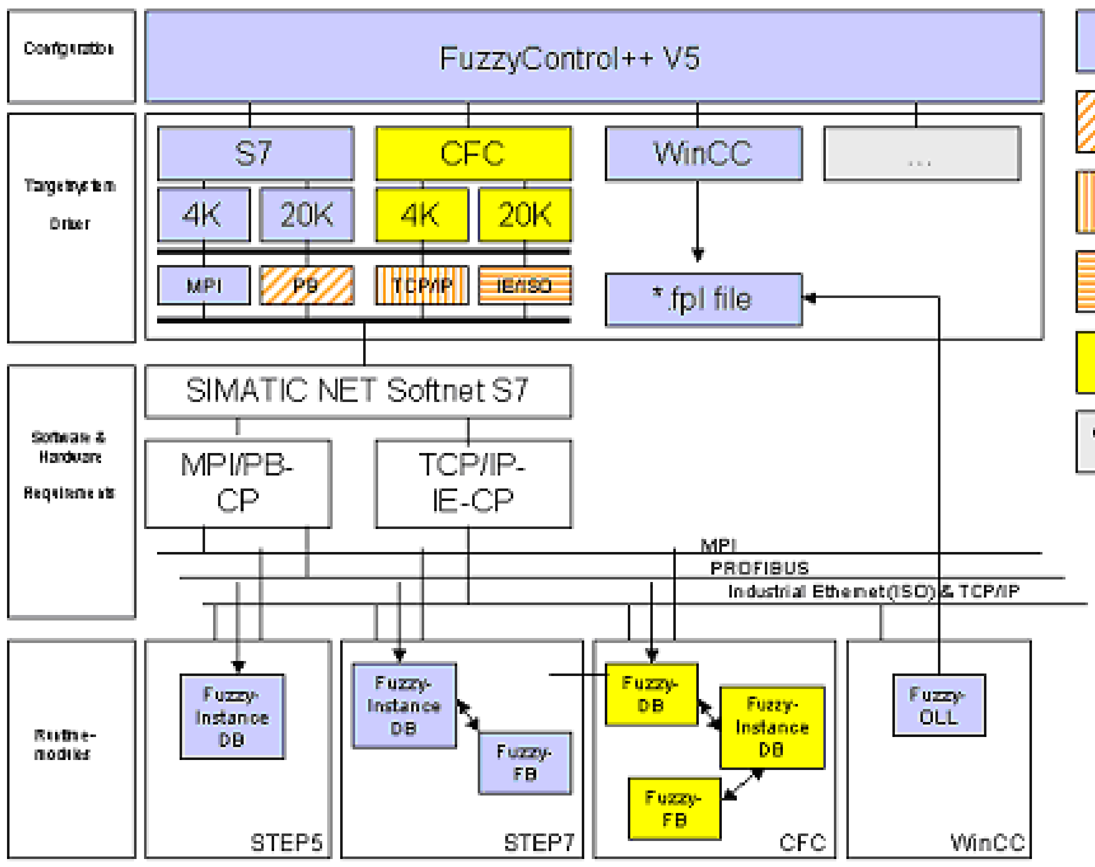
7.4 Menu: Targetsystem

Using the parameterized fuzzy system algorithm , the SIMATIC S7, SIMATIC CFC and SIMATIC WinCC targetsystem runtime modules calculate the associated output values from input values which can be defined online. If the targetsystem is a SIMATIC S7, the fuzzy system information is transferred to the targetsystem by writing it to a data block (DB). With SIMATIC WinCC, the fuzzy system is transferred to the runtime module (Fuzzy.OLL) in the form of an *.fpl file .

The "*Targetsystem Manger*" and "*Targetsystem Selection*" menu options are available for **all** targetsystems.

The "*Targetsystem Connect*", "*Targetsystem Disconnect*", "*Targetsystem Read*" and "*Targetsystem Write*" commands are **only** available for the S7 targetsystems.

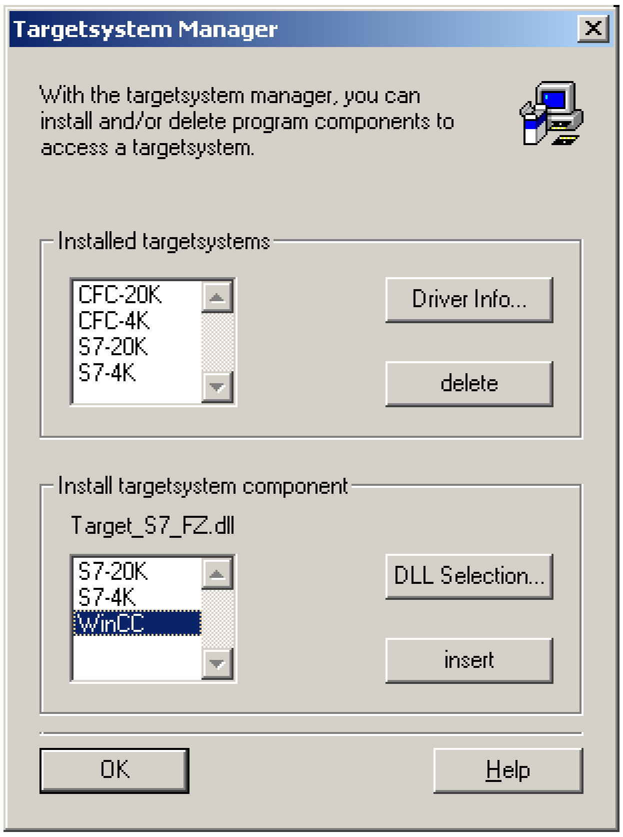
The following figure shows an overview of the interface, driver and targetsystem:



FuzzyControl++ V5 Drivers and Runtime modules

7.4.1 Manager

With the Targetsystem Manager you can obtain information about the installed targetsystems and special information about a chosen targetsystem. You can also install/reinstall further targetsystems.


Targetsystem Manager

With the targetsystem manager, you can install and/or delete program components to access a targetsystem.

Installed targetsystems

CFC-20K
CFC-4K
S7-20K
S7-4K

Install targetsystem component

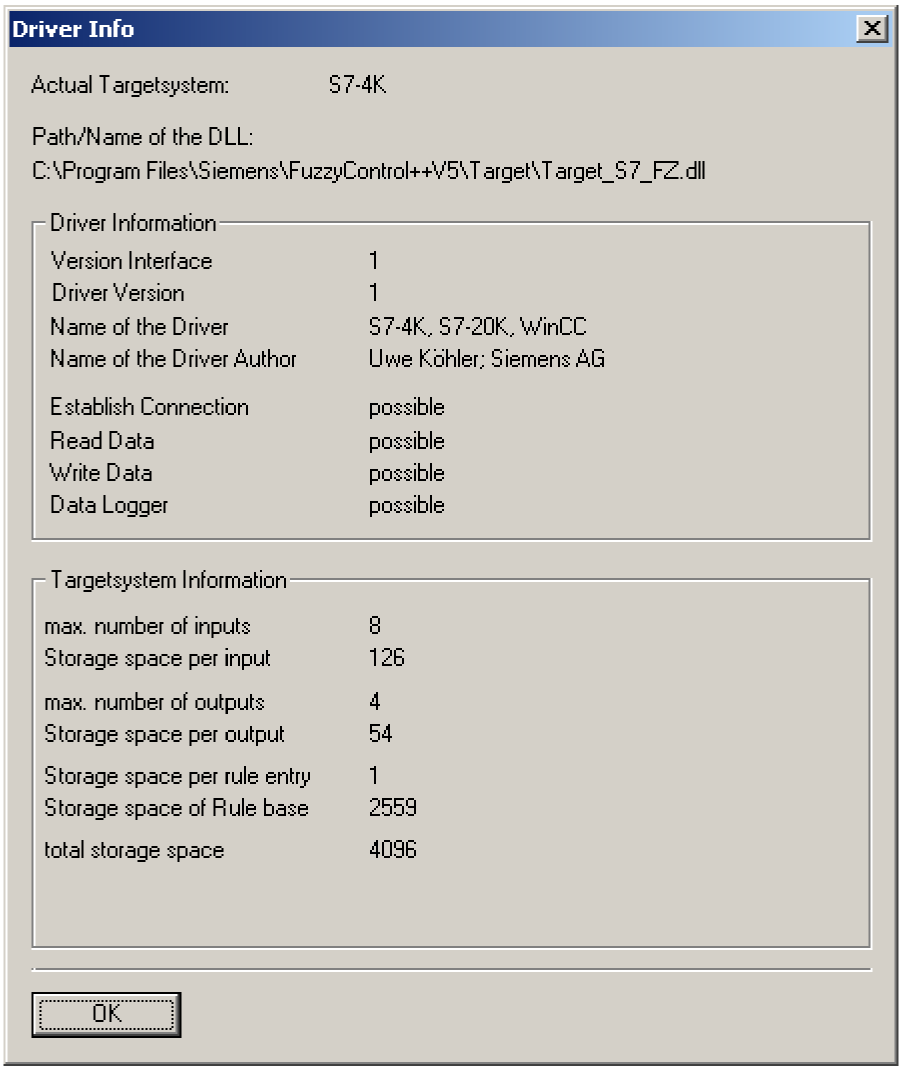
Target_S7_FZ.dll

S7-20K
S7-4K
WinCC

7.4.1.1 *Installed targetsystems*

In the targetsystem manager dialog box you find all installed targetsystems listed.

By choosing a listed targetsystem and pressing the *driver info* button you can see the information about the chosen targetsystem components in the *driver information* dialog box.



Driver Info

Actual Targetsystem: S7-4K

Path/Name of the DLL:
C:\Program Files\Siemens\FuzzyControl++V5\Target\Target_S7_FZ.dll

Driver Information

Version Interface	1
Driver Version	1
Name of the Driver	S7-4K, S7-20K, WinCC
Name of the Driver Author	Uwe Köhler; Siemens AG
Establish Connection	possible
Read Data	possible
Write Data	possible
Data Logger	possible

Targetsystem Information

max. number of inputs	8
Storage space per input	126
max. number of outputs	4
Storage space per output	54
Storage space per rule entry	1
Storage space of Rule base	2559
total storage space	4096

You also see the *driver info* dialog box when you create a new project and press the *driver info* button in the "new project" window.

The driver information dialog box provides general driver information and information about the quantity structure, e.g. the maximum number of inputs/outputs the targetsystem can work with.

At the processing stage of the project, NeuroSystems and *FUZZYCONTROL++* monitor these project limits. If there is a violation of these limits the user receives an error that says that the quantities given are invalid and that the targetsystem cannot proceed.

7.4.1.2 Deleting targetsystem components

To delete an unnecessary targetsystem you must choose the targetsystem in the menu *targetsystem-manager* and then click *delete*. This targetsystem will be deleted from both the targetsystem list (in the targetsystem manager dialog) and the registry.

7.4.1.3 Installing targetsystem components

To install a targetsystem you have to select a targetsystem DLL with the *DLL Selection* button. Following this, a menu appears where you must choose the targetsystem-DLL. After a DLL has been selected, targetsystems are offered in the list below, which are applied to the list above by via the *Insert* button and are made available for other dialogs such as *Choose Targetsystem* or *File New*.

The path/name of the DLL can be **reused** (for example when the S7-4K and S7-20K targetsystems are available within **one** DLL)

The WinCC, S7-4K and S7-20K targetsystems are supported as standard. These three targetsystems are covered by **Target_S7_FZ.dll**.

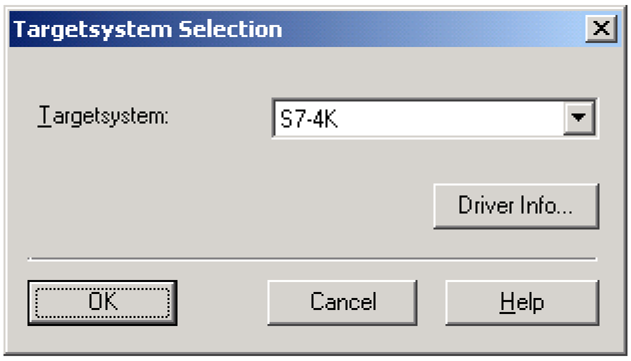
There is also the option of the CFC-4K and CFC-20K targetsystems for SIMATIC PCS7, covered by **Target_CFC_FC.dll**.

Caution: Because As entries are made to the registry during "install, delete or select targetsystem" , a user must have administrator rights so that the targetsystem settings can be applied. A "normal" user receives an error message.

7.4.2 Selection

With this command you can choose one of various targetsystems installed on which your fuzzy application should be run. The S7-4K, S7-20K, CFC-4K, CFC-20K and WinCC systems are available. Selecting the correct targetsystem avoids any possible memory and capacity problems when exporting the fuzzy systems to the targetsystems.

Note: **The CFC-4K and CFC-20K systems are optional.**



The following limits apply to the maximum possible number of inputs/outputs and rules for each targetsystem:

Targetsystem	**Maximum number of inputs**	**Maximum number of outputs**	**Maximum number of rules**
S7-4K, CFC-4K:	8	4	200
S7-20K, CFC-20K:	8	4	2000
WinCC:	8	4	2000

Note: **You must select the targetsystem before you configure the fuzzy system with the *File/New* command and do not change it afterwards if possible**, because some operations of *FUZZYCONTROL++* are implemented in a way which is specific to the targetsystem and problems could arise if you subsequently change it.

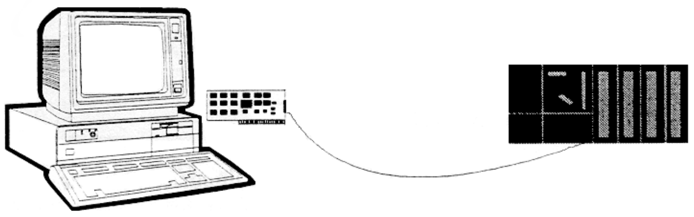
If you use *FUZZYCONTROL++* to configure a fuzzy smart object under SIMATIC WinCC, you must assign the *.fpl file created with *FUZZYCONTROL++* to the WinCC fuzzy object (Properties: fpl file) under SIMATIC WinCC. The "*Targetsystem/Connect*", "*Targetsystem/Disconnect*", "*Targetsystem/Read*" and "*Targetsystem/Write*" menu commands are not available for SIMATIC WinCC!

7.4.3 SIMATIC S7 and SIMATIC CFC Targetsystem

The S7-4K, S7-20K, CFC-4K and CFC-20K function blocks can be used here. The link with the SIMATIC is established via the **MPI** (Multi Point Interface) **card.** The SIMATIC NET SOFTNET configuration tool is also required. In addition, the S732.dll file is required, which is installed with the SIMATIC NET SOFTNET tool.

For the PGs that can be used, either the MPI interface is already integrated or a PC-MPI card must be used. Likewise, a PC-MPI card is required for the link with PCs.All cards are supported, up to the CP5511, which are also supported by the SIMATIC NET SOFTNET product.

With the help of an additional package and the relevant license, you can also use other communication networks in addition to MPI.



7.4.3.1 Connecting

This command establishes the connection with the data block of the selected targetsystem (S7-4K, S7-20K…). This offers the possibility of reading out or writing the DB (data block) belonging to the fuzzy FB (function block) of the programmable controller. For the SIMATIC S7-300 a 4 KB data block is available, and for the SIMATIC S7-400 a 20 KB data block is also available.

Caution: The 4 KB DB and the associated FB can also be used on the S7-400. For this, however, you must select the S7-4K/CFC-4K system as the targetsystem under *FUZZYCONTROL++* although the connection is actually being established with a SIMATIC S7-400!

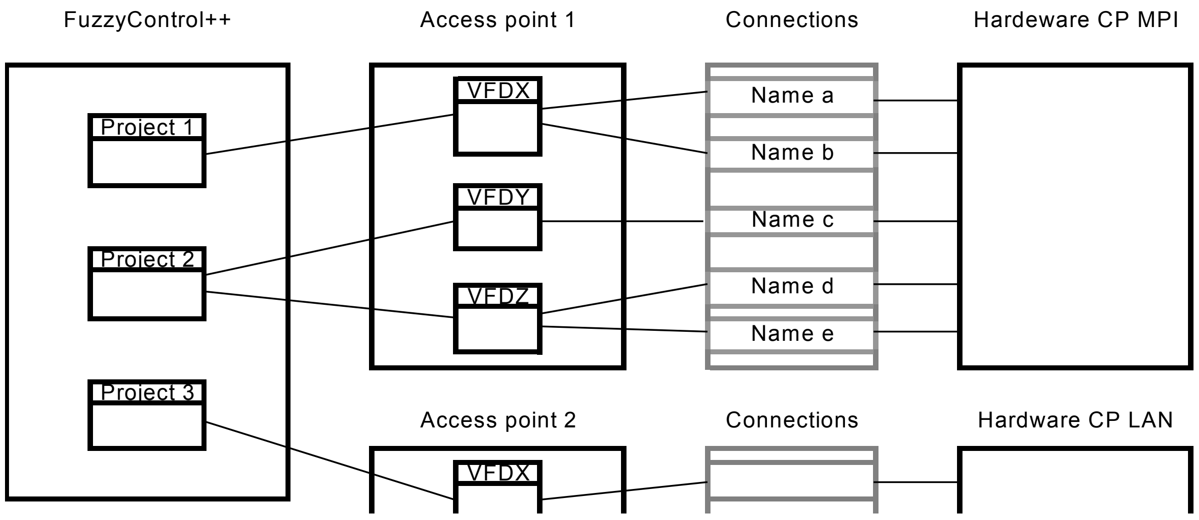
The following steps are used to establish a connection between the configuration tool and the SIMATIC S7:

1. On the targetsystem (SIMATIC S7-300 or SIMATIC S7-400) there is a fuzzy function block (FB30 or FB31) as well as at least one fuzzy data block (e.g. DB30).
2. The *FUZZYCONTROL++* configuration tool is opened and a project is edited.
3. The *Targetsystem/Connect...* command is activated.

4. You are requested to specify your connection in more detail in the *Connect* dialog field:

- First of all, you must state where the SIMATIC Communication Software "S732.DLL" is located on your computer. The preinstalled settings assumed that this file can be found in the system directory. If this DLL is in another directory on your computer, you can click on "Browse..." to find it or you can directly enter the full path.
- Choose the correct access point. The access points are associated with the CPs under the “set PC-PG interface” (SIMATIC NET). The corresponding CP is contacted over an access point.
- The VFD(Virtual Field Device) name in the “VFD-Name” field is automatically generated. The S7 connections are ordered by these VFD names.

If you wish to connect multiple projects with the targetsystem, all open projects require different VFD names!



MPI:

You must connect a SIMATIC S7 system via an MPI (multi point interface) card. For the communication itself you require the SIMATIC NET software - SOFTNET S7 PROFIBUS.

Enter the MPI-address of the CPU directly into the “*target address*” field. This is a set of three numbers (each between 0 and 126).

PB (Profibus):

You must connect a SIMATIC S7 system via a profibus-interface card. For the communication itself you require the SIMATIC NET software - SOFTNET S7 PROFIBUS.

Enter the MPI/Profibus-address of the CPU directly into the "*target address*" field. This is a set of numbers (maximum of 3, each between 0 and 126).

IE (Industrial Ethernet):

You must connect a SIMATIC S7 system via an ethernet card. For the communication itself you require the SIMATIC NET software - SOFTNET S7 IE.

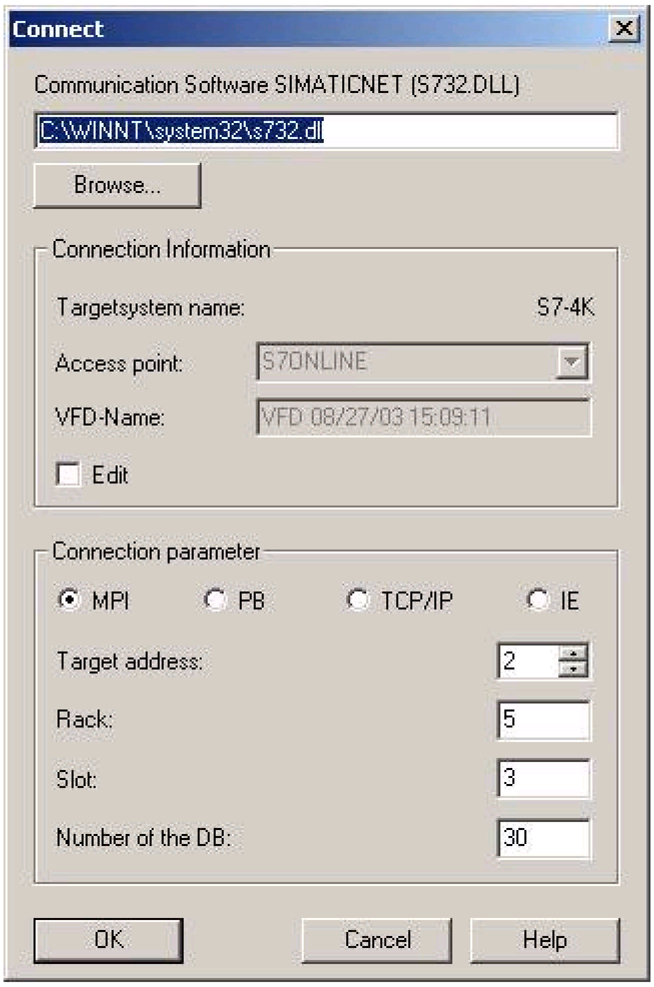
Enter the industrial ethernet-address directly into the "*target address*" field. This is a set pf 6 hexadecimal numbers between 0x0 and 0xFF, separated by dots (e.g. 00.FF.0A.F0.EE).

TCP/IP:

You must connect a SIMATIC S7 system via an ethernet card. For the communication itself you require the SIMATIC NET software - SOFTNET S7 IE.

Enter the TCP/IP-address directly into the "*target address*" field. This is a set of 4 3-digit numbers between 0 and 255, separated by dots (e.g. 141.8.10.237).

- "Rack input box:": Here you must enter the S7 rack (between 0 and 7).
- "Slot" input box: Here you must enter the slot used in the rack (between 0 and 18).
- "Number of the Data block" input box: Here you must enter the number of the fuzzy data block on the S7-CPU (between 0 and 999999).


Connect

Communication Software SIMATICNET (S732.DLL)

C:\WINNT\system32\s732.dll

Browse...

Connection Information

Targetsystem name: S7-4K

Access point: S7ONLINE

VFD-Name: VFD 08/27/03 15:09:11

☐ Edit

Connection parameter

MPI ● PB ○ TCP/IP ○ IE ○

Target address: 2

Rack: 5

Slot: 3

Number of the DB: 30

Note: Once a link has been established, you can not only read and write fuzzy data blocks but display and archive process data online in the configuration tool's curve plotter.

For further information on setting up the connection, please see the relevant SIMATIC-SIMATIC NET documentation.

If the connection is successfully made, the fuzzy system that is opened can be transferred to the data block in the CPU. To do this, the *Write* menu command is called (see chapter 2.6.3.4)

Layout of Project-access point-VFD-Connection-CP:

7.4.3.2 Disconnect

This command breaks a connection with the targetsystem (S7-4K, S7-20K …).

7.4.3.3 Read

This command reads out a data block from the targetsystem (S7-4K, S7-20K…). **During reading the current fuzzy system loaded in *FUZZYCONTROL*++ is overwritten with the DB read.** Therefore you must confirm that you wish to read in the DB in a pop-up dialog box. If you do not wish to overwrite your current fuzzy system, you must create a new, empty fuzzy system before reading in the DB and overwriting it with the DB data. The number of inputs and outputs, the rule base, etc., do not have to match those of the fuzzy system of the DB read in!

7.4.3.4 Write

This command writes the data to a data block in the targetsystem selected (S7-4K, S7-20K …). For this purpose the data block has to be available on the targetsystem. Make sure that processing of the fuzzy component is stopped during transmission!

•

7.4.4 SIMATIC WinCC Targetsystem

The fuzzy runtime module (implemented with "Fuzzy.oll") is an extension of the Siemens SIMATIC *WinCC* software. It allows the employment of modern methods and applications that make use of the capabilities of fuzzy systems. The *FUZZYCONTROL*++ PC configuration tool is used to configure the fuzzy systems.

To make the OLL-object available for use in the WinCC graphics editor "Graphics Designer" , you must proceed as follows:

- Open the context menu for the "Graphics Designer" (by clicking on "Graphics Designer" with the right mouse button and then selecting "Graphic-OLL".
- Select "fuzzy.oll" in the left-hand sub-window (available Graphic-OLLs) and move it to the right-hand sub-window (selected Graphic-OLLs) using the arrow button.

The "Graphics Designer" object list now contains the new block in the "Smart Objects" group. You can use the new object in the same way as the existing WinCC smart objects.

After you have created a new fuzzy object on the notepad of the "Graphics Designers" , you can define its behavior with the associated "Object Properties" window. You can open that window by right-clicking on the object and then selecting "Properties" or with a double-click of the left mouse button. You can assign fuzzy input/output characteristics to the WinCC object created by specifying an *.fpl file created with *FUZZYCONTROL++*. The file in question must be specified for the attribute "fpl file" (under "*Properties/Miscellaneous*"). It must be entered with its full pathname. If the file is read in successfully, the number of inputs and outputs (under "*Properties/ Miscellaneous* ") of the configured fuzzy system is displayed. The inputs and outputs are also indicated graphically by small lines on the object. Under "*Properties/Input/Output*" all the inputs and outputs are displayed with their current values. Fuzzy objects can have up to 8 inputs and 4 outputs.

You can perform a simple function test of the module by changing the input values manually ("*Object Properties*" window under "*Properties/Input/Output*") and observing the resulting change in the output values.

The individual inputs and outputs can be addressed via direct connections, variables or C actions or be linked with other objects (dynamization) . The new object type is therefore fully integrated into the WinCC environment.

For further information, please consult the relevant WinCC manuals.

7.5 Menu: Test

In the *Test* menu you can:

- Check the input/output characteristic of the fuzzy system
- Display the results graphically in various ways.

7.5.1 3-D Graphics

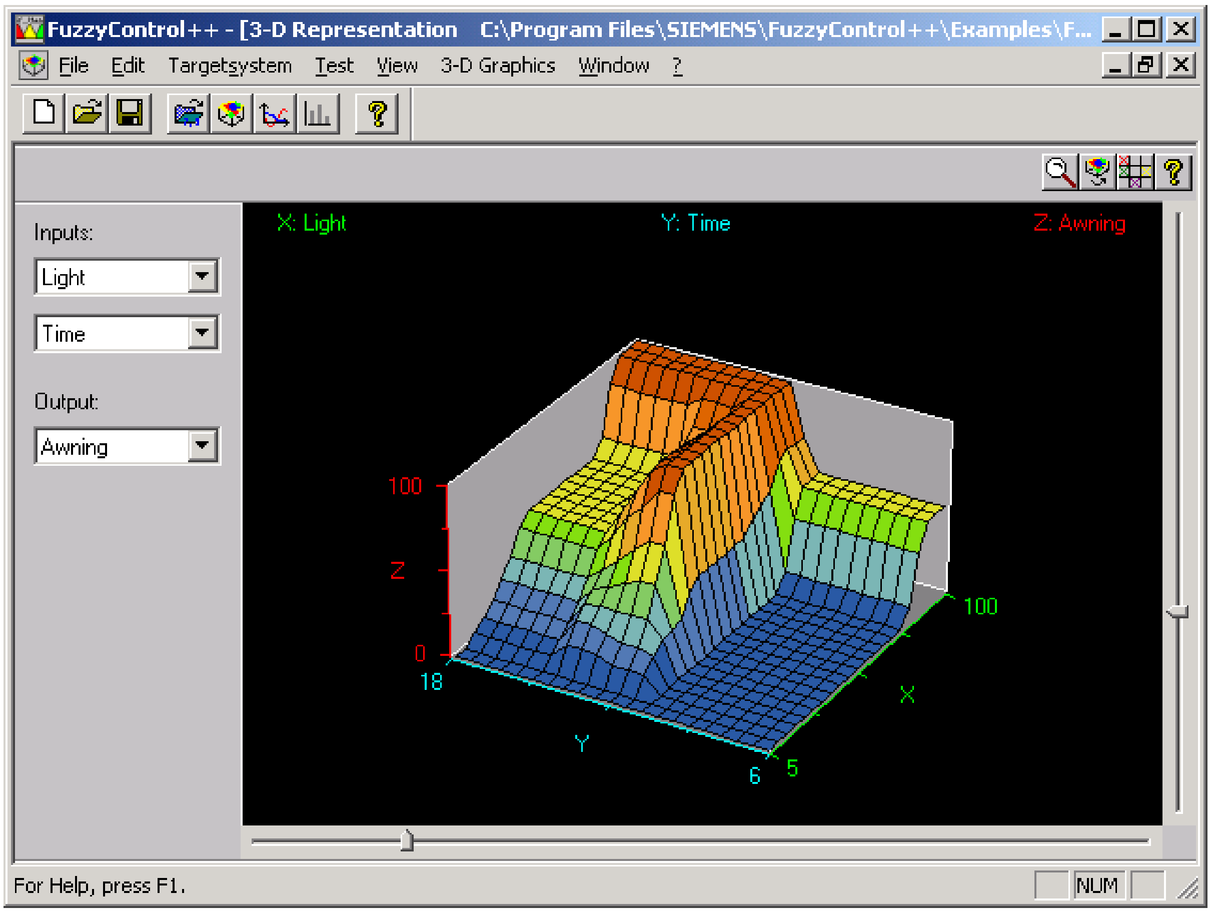
Shortcut: On the toolbar with LMB on the icon

or <ALT+T, D> key sequence

7.5.1.1 3-D Representation

With the *3-D Representation* it is possible to view the input/output characteristic of two inputs and one output of the fuzzy system in 3-D representation. The inputs are assigned to the X- or Y- axis, the output to the Z-axis.

The following figure shows the characteristic field that you can obtain for the inputs and output (*Awning*) of the *Awning* fuzzy system. The display range for the three axes were defined in the *Input/Output Properties* dialog box as the minimum and maximum values for the inputs and outputs in question. Normalization of the inputs and outputs affects the axis scaling in the 3-D display. The dot (in the figure front right) marks the minimum value of the parameters represented on the axes.


FuzzyControl++ - [3-D Representation C:\Program Files\SIEMENS\FuzzyControl++\Examples\F...
File Edit Targetsystem Test View 3-D Graphics Window ?
Inputs:
Light
Time
Output:
Awning
X: Light
Y: Time
Z: Awning
100
Z
0
18
Y
6
5
X
100
For Help, press F1.
NUM

You can view the characteristic field from various directions if you rotate the model horizontally and vertically using the sliders on the lower and right-hand edges of the display. You must click the slider and hold the LMB down while dragging it.

If the project has more than two inputs or more than one output, you can select and assign inputs or outputs to the axes in the selection boxes under *Inputs* or *Outputs*. In this way, it is possible to view the entire input/output characteristic of a project with different characteristic fields.

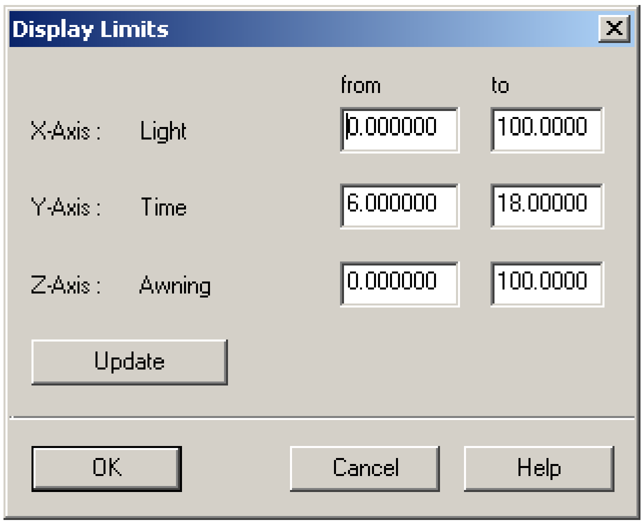
Note: If you click the 3-D graphic window with the RMB you can copy the graphic onto the Windows clipboard and print it out or process it further using other Windows applications.

7.5.1.2 *Display Range*

Shortcut: In the *3-D Representation* window

On the toolbar with LMB on the icon

You can define the display range of the 3-D Representation. As standard the min and max values of the inputs and outputs will be shown.



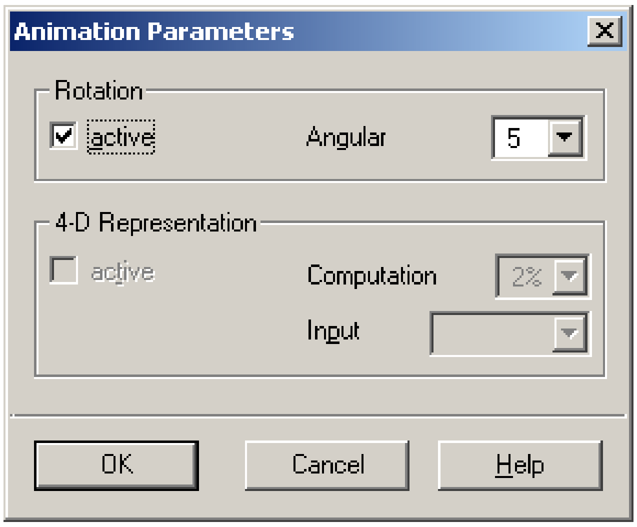
Display Limits

		from	to
X-Axis :	Light	0.000000	100.0000
Y-Axis :	Time	6.000000	18.00000
Z-Axis :	Awning	0.000000	100.0000

7.5.1.3 *Animation*

Shortcut: In the *3-D Representation* window

Starting:	Press the button
or	Click with the RMB and select *Animation on/off*
Stopping:	Press the button
or	Click with the RMB and select *Animation on/off*



The animation function of *FUZZYCONTROL++* permits automatic changes in the display parameters of the 3-D graphic:

- Rotation of the characteristic field about the vertical axis and
- Change in the characteristic field in accordance with a fourth parameter that runs through its defined range ("4-D Representation")

After you have activated animation, the *Animation Parameters* dialog box appears with which you can organize the *Rotation* and the *4-D Representation*:

Rotation

You can select the function by clicking on the associated *active* button. You can set the increments for rotation by entering the *Angular Increment*. To do that, select an angle specified in the entry field (1° to 45°). The larger the angle selected, the faster rotation will be. Rotation begins after you have closed the *Animation Parameters* window with *OK*.

Note: In the *3-D Representation* window the button is now replaced by the button with which you can stop rotation.

4-D Representation

This function of *FUZZYCONTROL++* allows you to view the influence of a third input signal on the input/output characteristic of the fuzzy system. The characteristic field of the 3-D display is repeatedly calculated and updated for successive values of this input signal, considered to be a parameter. In this way, you can obtain an indirect impression of the "fourth dimension" from the sequence of characteristic fields. The sequence is repeated cyclically.

You select the function *4-D Representation* by clicking the *active* button. (Of course, this is only possible if the project has more than two input signals.) Under *Input* you have a selection table from which you can pick the "third" (parameter variable) from the "remaining" inputs. Because you can select any two inputs in the *3-D Representation* window for the 3-D display, you can also select any input signal as the "third" input signal for the 4-D display function.

You can set the increments for the change of parameter value in *Calculation Step*. To do that, select a percentage value in the entry field. It then uses this portion of the whole range to distinguish the successive characteristic fields. For example, if you select 5%, an animation cycle contains 20 steps and 21 characteristic fields. If you select a greater percentage, the animation runs faster and with greater increments. The animation begins after you have closed the *Animation Parameters* window with *OK*.

Note: In the *3-D Representation* window, the button is now replaced by the button, with which you can stop the cyclic animation.

If you set both *Rotation* and *4-D Representation* to *active*, you can have the animation of the 4-D Representation rotate as it runs.

7.5.1.4 Setting the parameters (Graphical Representation)

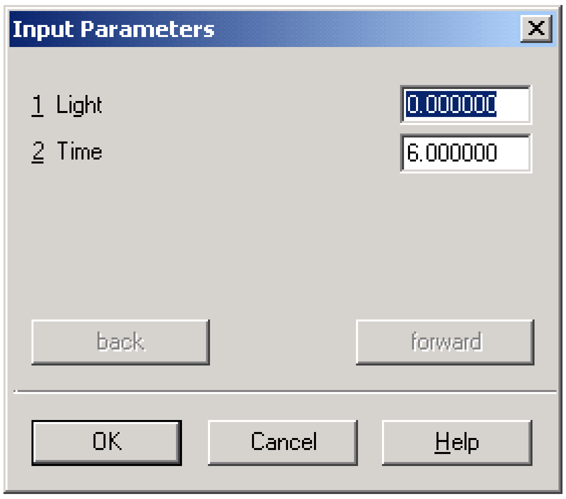
Shortcut: In the *3-D Representation* window

Activate the button

or Click with RMB and select *Parameterize Inputs...*

In many projects, more than three inputs are used. If you want to include their influence on a certain output signal in the display, you can do that by assigning certain numeric values to those inputs.

After activation, the following dialog box called *Inputs Parameters* appears in which you can enter the parameter values.



1 Light 0.000000
2 Time 6.000000

In this dialog box, all eight inputs are displayed by using *back / forward* but you can only change those values that really exist in the project. You can assign any value (up to 7 digits) to each input.

Apart form the two "running" parameters of the input values on the X/Y axis, the values assigned to the other inputs are taken into account directly in the graphical display. If you change an input assigned to the X- or Y-axis, the new graphic output is based on the parameter value of the input removed.

After confirmation with *OK,* the output characteristic field, which is valid for the parameters set, is calculated and displayed.

Note: After you have stopped a 4-D Representation you can read the associated value of the parameter variable ("third" input signal) for the "frozen" surface by activating the Parameter button and selecting this input. This makes selective evaluation of the influence of the third input signal possible.

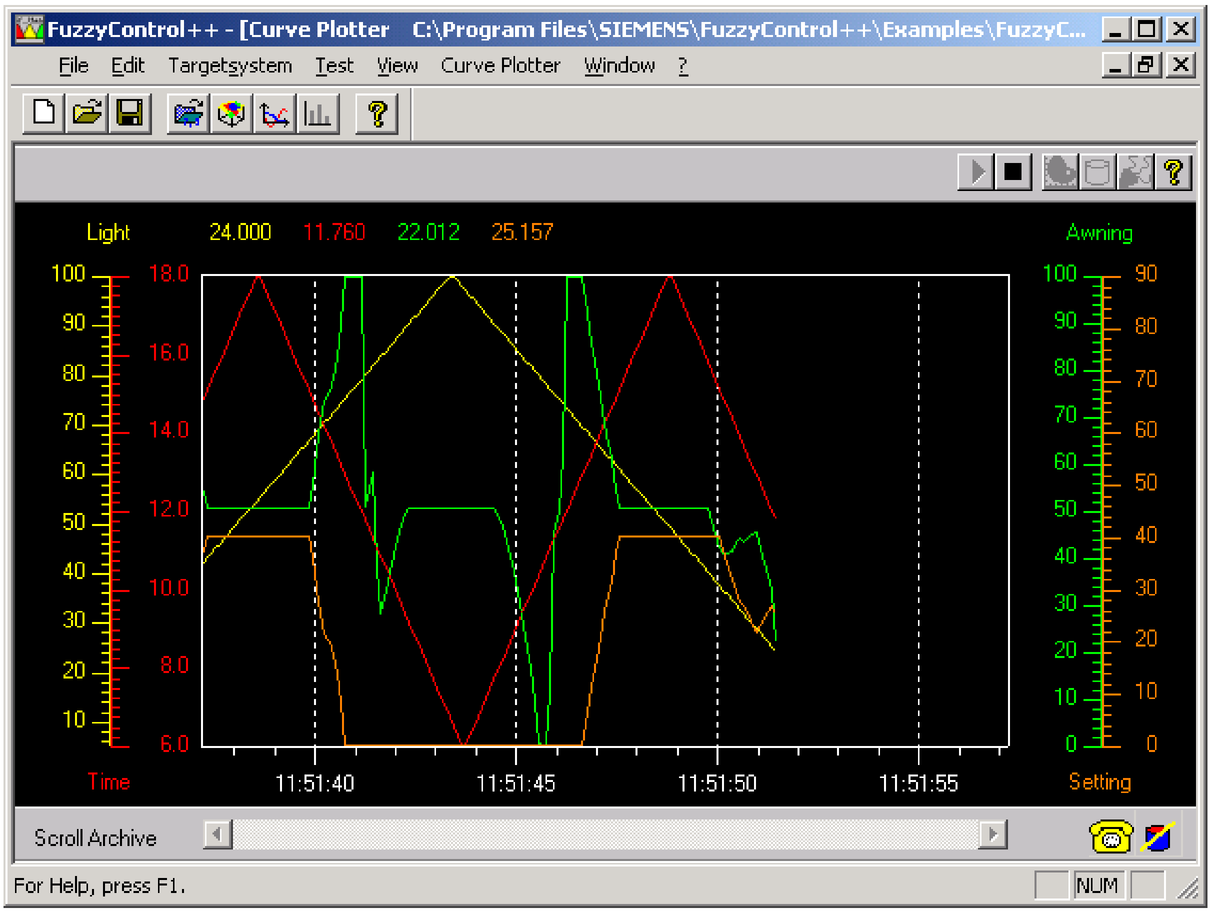
7.5.2 Curve plotter

Shortcut: In the toolbar with LMB on the icon

or <ALT+T, C> key sequence

The curve plotter permits graphical display of up to any four input- or output- signals against time. The time axis has a length of 10 to 9999 seconds. The display is in real time and you can observe its progress by the hh:mm:ss coordinate values. You can also archive the curve plotter data on hard disk and load it again as required. A curve generator can generate input signals, which makes it possible to test the fuzzy system and view the results graphically.

Note: The change against time of the input values provides a sort of "cross-section" through the characteristic field for a fuzzy system, which is known to be static. The setting of the time parameters therefore affects the resolution of this cross-section.


FuzzyControl++ - [Curve Plotter C:\Program Files\SIEMENS\FuzzyControl++\Examples\FuzzyC...
File Edit Targetsystem Test View Curve Plotter Window ?
Light
24.000
11.760
22.012
25.157
Awning
Time
11:51:40
11:51:45
11:51:50
11:51:55
Setting
Scroll Archive
For Help, press F1.
NUM

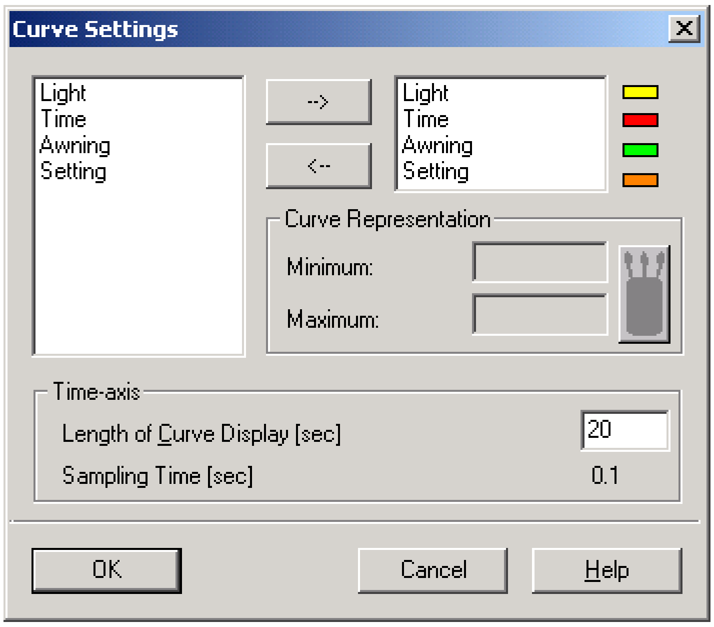
7.5.2.1 *Opening the curve plotter*

You can open the *Curve Plotter* window by activating the Curve Plotter menu command or by clicking on the curve plotter icon on the toolbar.

7.5.2.2 *Selecting the inputs and outputs to be displayed in the curve plotter*

Shortcut: In the *Curve Plotter* window: Press the button

In a curve diagram, you can display up to four inputs or outputs as curve traces (each in a different color). In the left hand side of the window you will see a list of all the inputs and outputs in the project, and on the right, a list of the variables currently displayed in the curve plotter window with the colors assigned to them. You can define the right-hand list as a selection of up to four signals from the left-hand list. You can do that by selecting the variables in the lists and then clicking the "arrow buttons", but it is also possible to double-click on a signal in the left-hand list to place it in the right-hand list and to double-click on a signal in the right-hand list to remove it again from the list of signals to be displayed in the curve diagram.


Curve Settings

Light
Time
Awning
Setting

Light
Time
Awning
Setting

Curve Representation
Minimum:
Maximum:

Time-axis
Length of Curve Display [sec] 20
Sampling Time [sec] 0.1

If you want to assign a certain color to a variable from the right-hand list, select this input/output.

You can select the basic colors and user-defined colors by using the button.

The display range is always selected automatically as it is defined in the *Edit* window of the corresponding inputs and outputs (*Maximum and Minimum* values).

If you want to change the curve scaling, you must select the corresponding parameter from the right-hand list and manually enter the maximum and minimum limits for the range of values to be displayed.

7.5.2.3 *Adjusting the length of the time axis of the curve plotter*

Shortcut: In the *Curve Plotter* window: Activate the button

A time scale is shown along the bottom of the diagram window, which is scaled to match the length of the curve display specified by you. You can set the length of the curve display by entering a value in the field *Length of Curve Display* in the *Curve Settings* window. You can select a shorter or longer time axis to match the required time resolution for displaying events. Values between 10 and 9999 seconds are possible. 150 values per curve are displayed and linear interpolation is used between these values. The sampling time is calculated from the selected length of the curve and the 150 values that can be displayed and it is shown in the *Curve Settings* window. It must be at least 0.1 s.

7.5.2.4 *Starting and stopping recording*

Shortcut: In the *Curve Plotter* window:
Start: Press the ▶ button
Stop: Press the ■ button

You can start recording by pressing the *GO* button. Recording progresses from right to left and is updated in the same time-period as the sampling time. When the end of the diagram window is reached, the entire curve display is shifted to the left by a quarter of the diagram length. During recording, the current values of the signals entered are displayed in the upper part of the diagram in the curve color. If you press the *STOP* button, recording is stopped.

7.5.2.5 *Archiving the recording data in memory*

After the end of recording, not only the currently displayed section of the curve (corresponding with the *Length of the curve display*), but also a section to the left of the current section is available. Archiving the values permits retrospective analysis of the behavior of the fuzzy application. The curves are archived in the RAM of the computer up to a length of about 500 sampling values. All "previous" values are cut off.

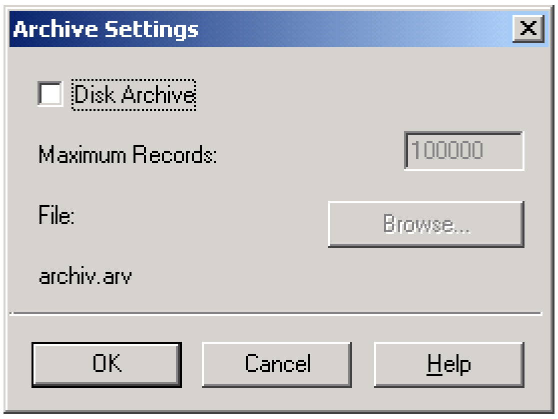
7.5.2.5.1 The hard disk archive

Shortcut: In the *Curve Plotter* window: Activate the button

The option of archiving curve plotter data on hard disk has the following advantages:

1. You can store more data than is possible in the RAM of the computer.
2. Permanent storage of data is possible.
3. You can retrieve and analyze the data later on.
4. You can use the archived data directly as test data.

If you want to archive the data on the hard disk, you must open the *Archive Settings* window **before you start recording**. In the *Archive Settings* window, you must check the *Disk archive* option and enter the maximum number of records. The number of records must be at least 10 and no more than 100,000. The data is stored in an *.arv file (archive file). With *Browse* you can either open an existing *.arv file or create a new archive file.



Archive Settings

☐ Disk Archive

Maximum Records: 100000

File: archiv.arv

When you open an existing archive, the stored data is displayed in the curve plotter diagram. The limits of the value ranges are automatically adapted to the archive data.

If you start recording again, a query is displayed whether you want to overwrite and also delete the existing archive data.

7.5.2.5.2 Archive analysis

This section applies both to archives in the RAM and those on the hard disk.

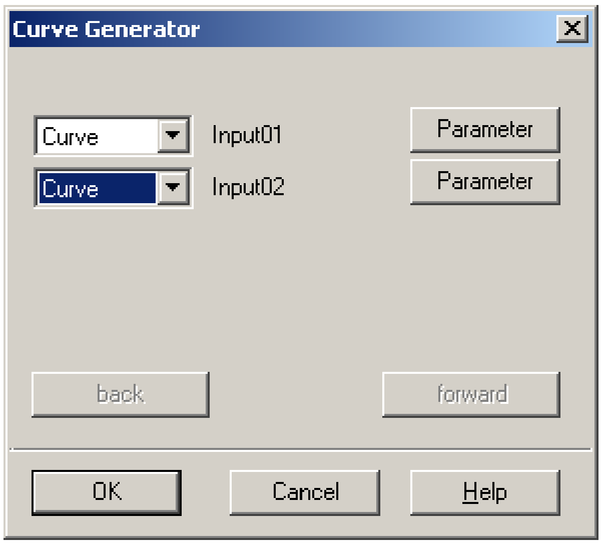
The scroll bar under the diagram is used to view the archived parts of the curves ("scroll through the archive "). If you click the right and left arrows, the diagram is scrolled in small steps. It is scrolled in large steps if you click between the arrows and the slider. Fast positioning by "dragging" the slider is also possible.

Use of the vertical read-off line, which appears after you stop recording, permits a precise analysis: Using the mouse, you can place it in any position in the displayed section of the curve. At the same time, the upper part of the diagram displays the values of the recorded signals that are valid at the position of the read-off line.

During analysis of the archive it is also possible to change the inputs and outputs displayed. If you select another input or output signal, the curves and the numeric values are updated.

7.5.2.6 Curve generator - Parameterization of the curves

Shortcut: In the *Curve Plotter* window: Press the button



To test the input/output characteristic of the configured fuzzy system you can stimulate the inputs with different signals using the **curve generator** and observe and analyze the response of the network with the curve plotter. You can set the input curves by activating the *Curve*

Generator button. Stimulation with constant or triangular signals is possible. The default setting for all inputs is a *constant* progression over time with the value 0.0.

If you activate the arrow button on the display window of an input, you can set the following values:

- **Constant:**

  After you have entered a numeric value and confirmed with *OK* the numeric value is accepted and displayed.

- **Maximum:**

  To assess the effect of the input signals at their value range limits, you can set the input in question to the maximum value of the value range using the *Maximum* menu option. This value is the *Maximum* set in the *Edit Input* window.

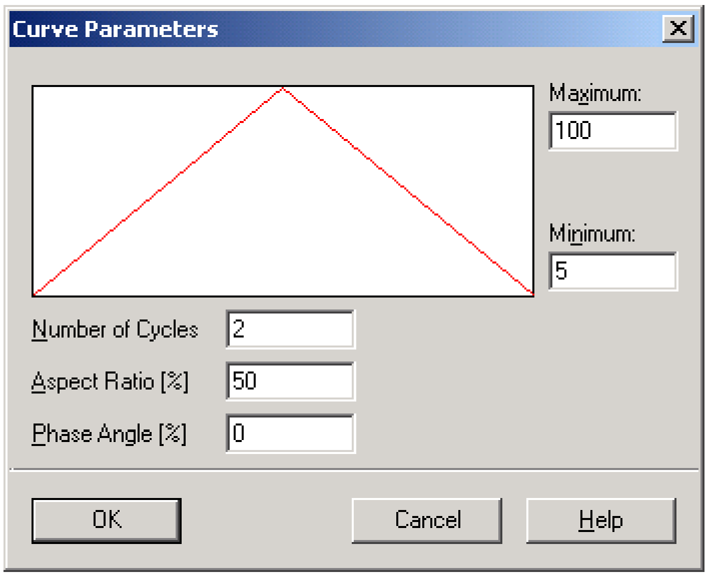
- **Minimum:**

  Similarly, you can apply the minimum value of its value range to the input by selecting *Minimum*. This value is the *Minimum* set in the *Edit Input* window.

- **Zero:**

  If you select *Zero menu option* you can set the input concerned to zero.

- **Curve:**



Maximum: 100
Minimum: 5
Number of Cycles: 2
Aspect Ratio [%]: 50
Phase Angle [%]: 0

If you select the Curve menu option, the input is stimulated with a triangular curve.

In the *Curve* window, you can influence the shape of the curve as follows:

1. Define the amplitude of the test signal by entering the values for *Maximum* and *Minimum*. If these values are the same as the range limits of the input signal, the period function just touches the top and bottom horizontal borders of the window, but smaller and larger values are also possible.
2. Enter a value for the number of cycles. This specifies how many times the triangular test function is inputted during the length of the curve display. The larger this value is, the steeper the edges of the triangles will be.
3. The value of the aspect ratio is the percentage ratio of the duration of the rising edge of the triangular signal to the total duration of the period. The default setting is 50%.
4. The phase angle defines the starting point of the test signal with respect to the curve display in the diagram. The display in the *Curve* window shows at what phase angle recording will begin. A phase angle of e.g. 25% offsets the test signal in the window by 25% of the cycle length to the left in the curve window. Curve recording therefore starts at 25% of the cycle length. The default setting is 0%.

After each change in the parameters, the test signal trace is recalculated and updated in the graphical display of the window. If you press the *OK* button, the parameters set are stored and the *Curve* window is closed. In the *Curve Generator* window, *Curve* is displayed instead of the numeric value for the input concerned.

You can not only enter the curve parameters for the inputs, but also display the resulting values of the outputs in the *Curve* window. This display only works if constant values are assumed for the inputs. If *Curve* is selected for an input, the "*???*" expression is displayed for all outputs.

7.5.2.7 Online/Offline

If there is an online-connection, a red telephone will be displayed. I.e. after *targetsystem/connect* the online files from the controller (SIMATIC S7-300, SIMATIC S7-400) can be displayed and recorded.

An offline connection is shown by a yellow telephone.

7.5.3 Rule Activity

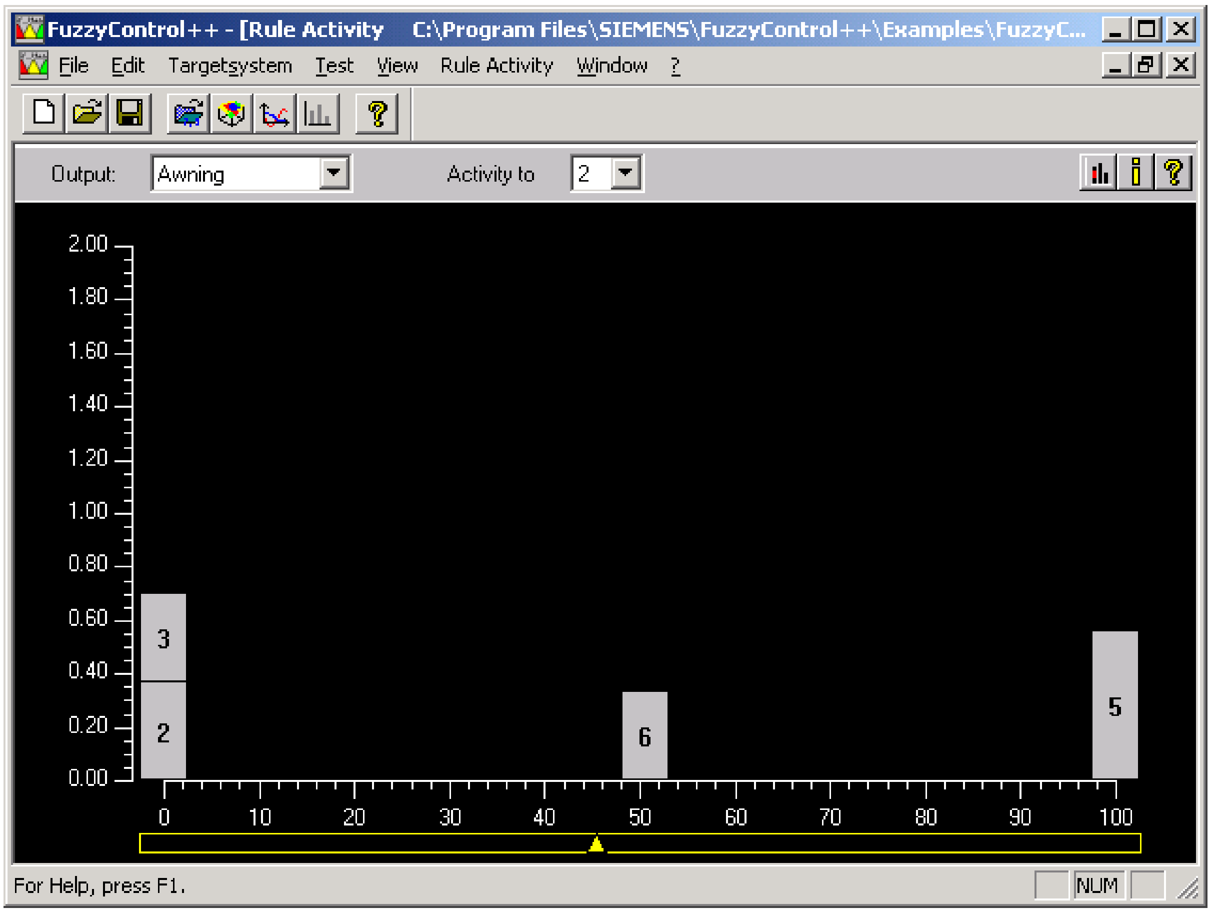
Shortcut: On the toolbar with a click on the icon

<ALT+T, R> key sequence

This menu option opens a window for graphical display of the rule activity for the outputs and illustrates defuzzification according to the centroid method.

Note: Before you open the *Rule Activity* window, the window for the *curve plotter* should already be open. The two windows must be seen as interconnected because the rule activity is only displayed as the curve plotter records.

The example in the following figure is intended to explain the operating elements of this window.


FuzzyControl++ - [Rule Activity C:\Program Files\SIEMENS\FuzzyControl++\Examples\FuzzyC...
File Edit Targetsystem Test View Rule Activity Window ?
2.00
1.80
1.60
1.40
1.20
1.00
0.80
0.60
0.40
0.20
0.00
3
2
6
5
0
10
20
30
40
50
60
70
80
90
100
NUM

Output: Awning

Activity to 2

For Help, press F1.

Axis description

The horizontal axis of the *rule activity* diagram represents the value range of the output signal from which the non-fuzzy value is calculated by defuzzification. The vertical axis shows the activity of the rules (rule result). Values larger than 1 can occur due to the SUM-MIN inference used in the fuzzy system. You can set the range displayed to the values 1, 2, 5 and 10 using the selection field *Activity to*. The default value is the range up to 2.

Diagram content

The diagram shows the activity of each rule in the form of a bar. The figures inside the bar indicate the rules that are currently active. The height of the bar corresponds to the rule activity and the position of the bar to the position of the membership functions (singletons).

The display below the X-axis indicates the non-fuzzy value of the output signal belonging to the current non-fuzzy input values. In conjunction with the columns above that, you can get a visual impression of the centroid method of defuzzification by imagining a "fulcrum" on which the columns balance.

Selecting the output

You can set and alter the output for which the rule activity is to be displayed using the selection field *Rule activity for*.

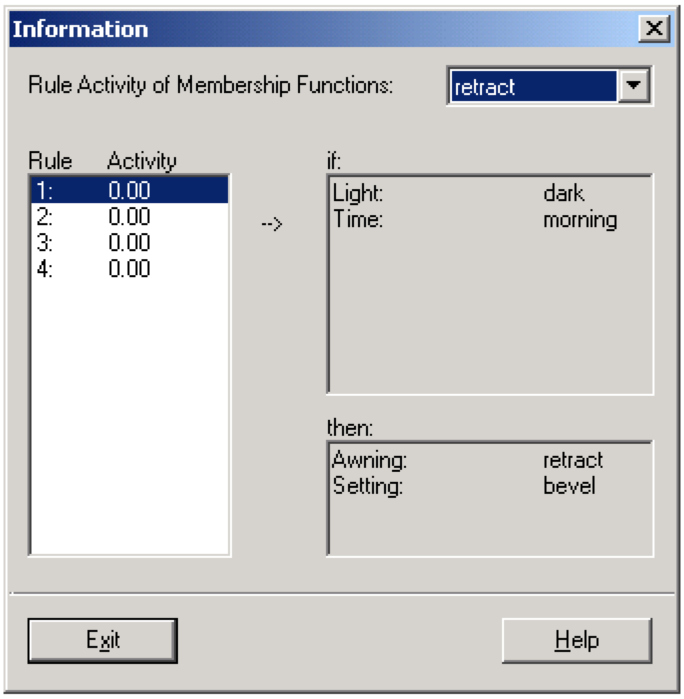
Information

There are two ways of displaying information about a membership function of the output.

1. In the diagram by clicking the corresponding bar with the RMB and selecting *Information*.
2. By clicking the i button and selecting the membership function from the *Membership functions* dialog box which then opens.

Both methods will enable you to open the *Information* window.

In this window the linguistic value (name of the singleton) of the selected output that is addressed by the rule is displayed under *Rule activity of the membership function*. Moreover, in the *Rule Activity* display field, all the rules are listed each with its degree of fulfillment in which the linguistic value of the selected output occurs. For the rules marked in this window, the corresponding linguistic values of the inputs (if) and the outputs (then) of the rules are displayed in the *if:* and *then:* fields.


Information

Rule Activity of Membership Functions: retract

Rule	Activity
1:	0.00
2:	0.00
3:	0.00
4:	0.00

-->

if:

Light:	dark
Time:	morning

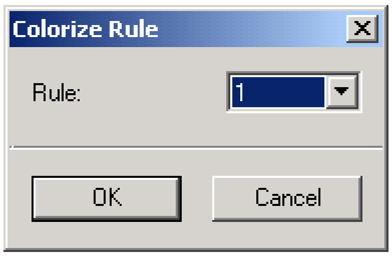
then:

Awning:	retract
Setting:	bevel

Coloring rules

In the diagram you can color a rule to identify it more easily.

You can do that either by clicking the icon or by clicking with RMB on the surface of the diagram and selecting *Colorize Rule*.


Colorize Rule

Rule: 1

This dialog box is then opened. In this dialog box you can enter and select the rules you want to color in the selection field. If you click *OK*, the selected rule is colored red.

You can use this dialog box to remove the color from the rule by selecting the item *none*.

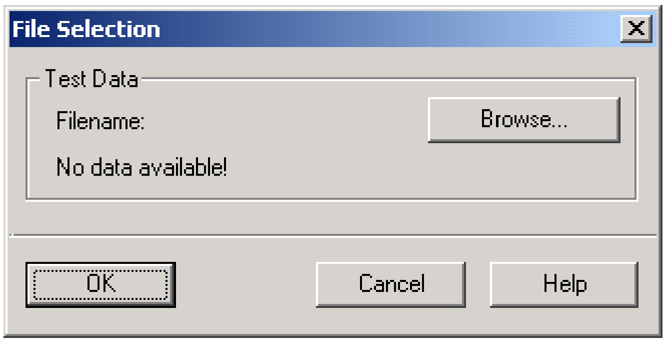
You can also remove the color from the rule by clicking on the diagram with RMB and selecting *Decolorize rule*.

7.5.4 File selection

Shortcut: Key sequence <ALT+T, F>

This option allows you to load a **test data file**. This file has the same structure as a learning data file of *NEUROSYSTEMS* but differs in that it can be used to test the properties of the fuzzy system. The results of the tests performed with the test file are displayed with the options *Vector Representation* and *Signal Representation* in the *Test* menu.

If you click on *File Selection* in the *Test* menu, the *Test File Selection* dialog box appears.



If you activate *Browse* you can load a **.dat* ASCII file from the *Open* dialog box, which then appears. After it has been loaded, its name is displayed in the dialog box *Test File Selection*. After you close the window with *OK* the file is used as the test file for all further work with *FUZZYCONTROL++*. Until you have loaded a test file into the current product, you cannot select the *Vector Representation* and *Signal Representation* menu options.

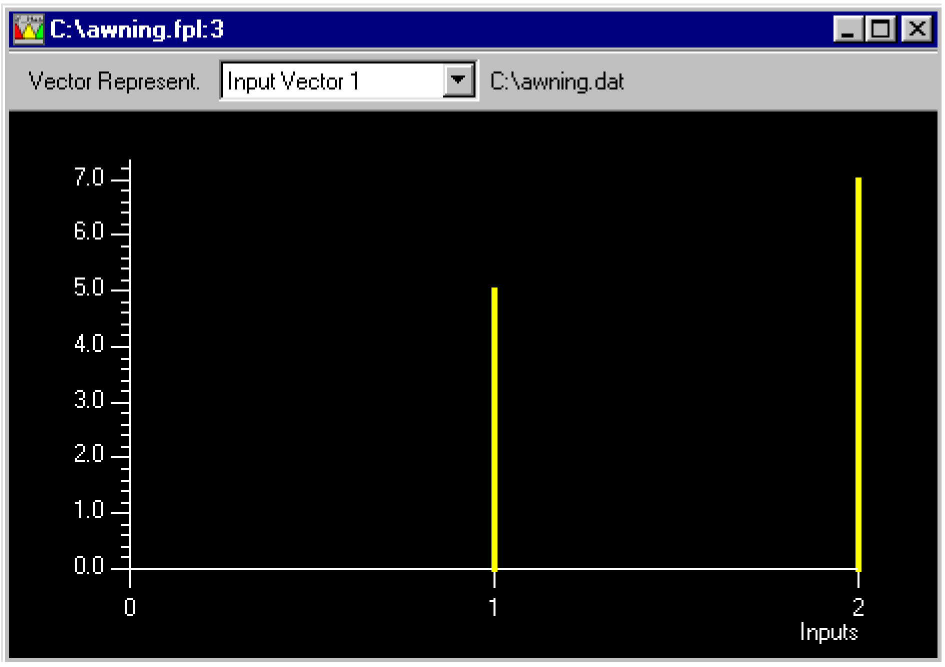
7.5.5 Vector Representation

Shortcut: Key sequence <ALT+T, V>

In the *Vector Representation* window, you can have either an input vector or an output vector of the test file displayed. If you have output vectors of the test file displayed, the output vector calculated by the fuzzy system is also displayed. The output vector of the test file is displayed in *yellow* and the output vector generated by the fuzzy system as a response to the test input vector is displayed in *red*. This allows you to judge to what extent the input/output characteristic of the fuzzy system matches the input/output characteristic represented by the test file.

The following figure shows the first input vector of the *awning.dat* test file with a "non-fuzzy" value of 5 for the "light" input variable and a value of 7 (o'clock) for the "time of day" variable. (Please note that the numbering of the vector components begins with 1, whatever names have been chosen for the inputs, e.g. Input00, Input01, etc.)

You cannot activate *Vector Representation* if you haven't selected a file with *File Selection...*.



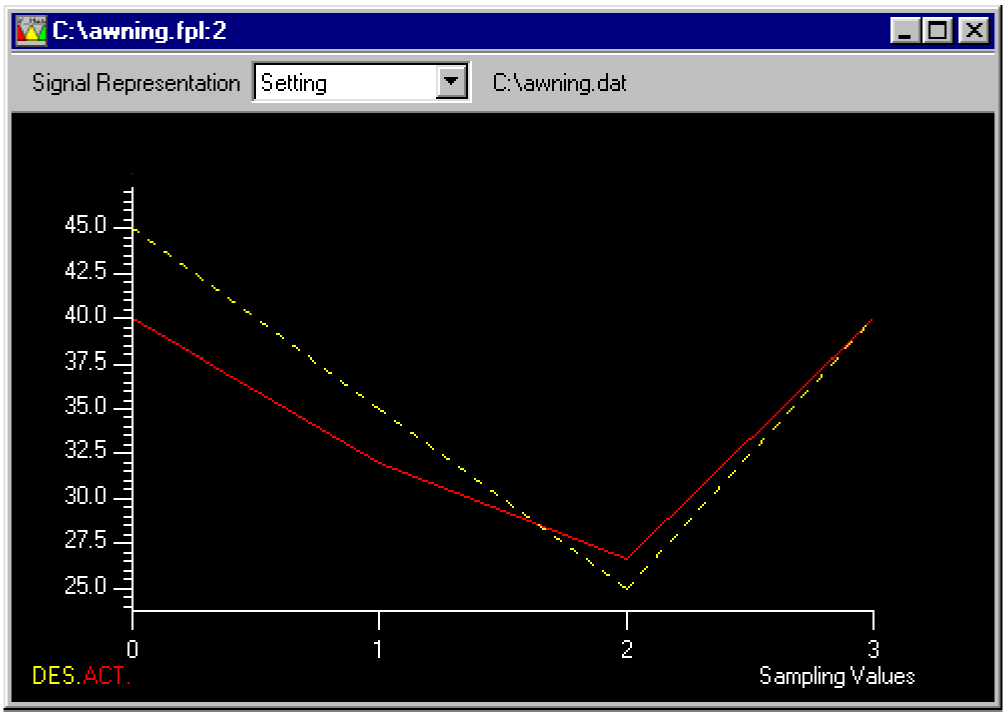
Note: If you click the display with *RMB,* a selection box opens. By selecting *Copy to Clipboard* you can copy the diagram into the Windows clipboard and print it or process it further with other Windows applications. If you select *Save data in file* it is possible to save the values of the outputs in a *.dat file (ASCII format).

7.5.6 Signal representation

Shortcut: <ALT+T, S> key sequence

In the *Signal Representation* window, you can graphically display the progression against time of either one input signal as defined in the test file or one output signal as calculated by the fuzzy system. The signal trace is interpolated linearly between the values for the points in time (sampling values). In the representation of an output signal of the fuzzy system, you can also display the output signal in the test file. The test output signal is displayed in *yellow* and the output signal generated by the fuzzy system as a response to the test input signal is displayed in *red*.

This allows you to judge to what extent the input/output characteristic of the fuzzy system matches the input/output characteristic represented by the test file.


C:\awning.fpl:2

Signal Representation: Setting

C:\awning.dat

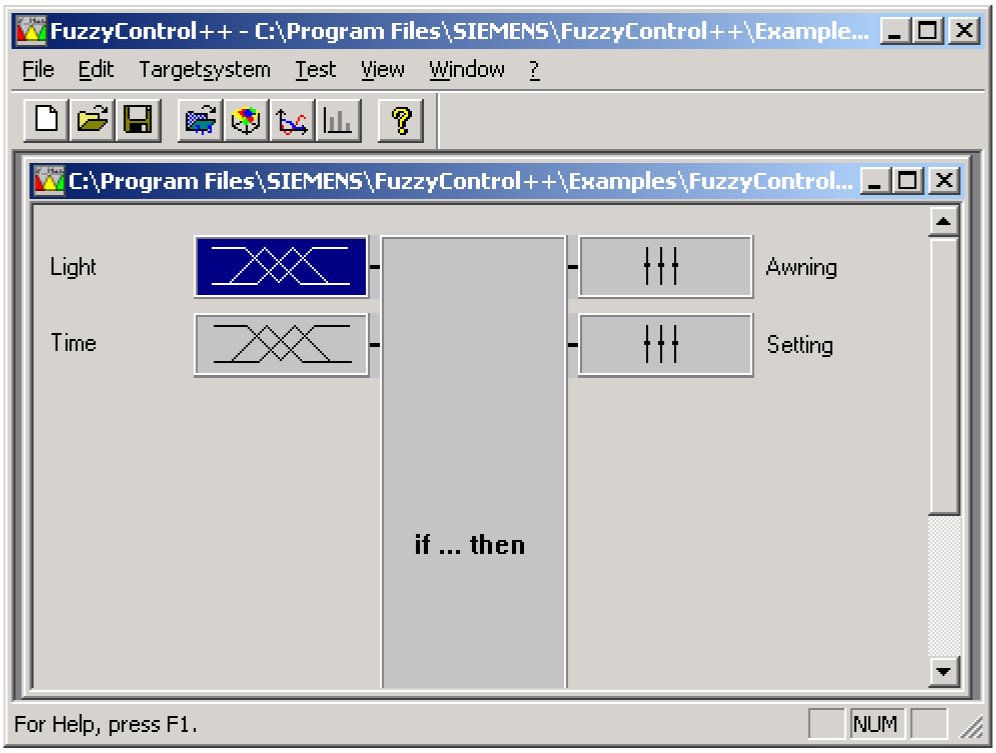
The figure shows the trace of the output in accordance with the input values contained in the *awning.dat* test file. The red line shows the output signal of the fuzzy system and the broken yellow line is the output signal of the test file.

You cannot activate *Signal Representation* if you haven't selected a file with *File Selection...*.

Note: If you click the display with *RMB*, a selection box opens. By selecting *Copy to clipboard* you can copy the diagram into the Windows clipboard and print it or process it further with other Windows applications. If you select *Save data in file* it is possible to save the values of the outputs in a *.dat file (ASCII format).

7.6 Menu: View

In the *View* menu you can show and hide the *Toolbar* and the *Status Bar* ("Press F1 to...") by clicking them with the LMB.


FuzzyControl++ - C:\Program Files\SIEMENS\FuzzyControl++\Example...

C:\Program Files\SIEMENS\FuzzyControl++\Examples\FuzzyControl...

For Help, press F1.

7.6.1 Toolbar

The toolbar is displayed if a checkmark is shown in front of *Toolbar* in the *View* menu. It is positioned horizontally at the top of the working window underneath the menu bar and contains the buttons for the most frequently required commands of *FUZZYCONTROL++*.

The toolbar provides quick access to the program options of *FUZZYCONTROL++* simply by clicking with the LMB.

Icons and functions of the toolbar

Opens the *Define Project* dialog box to create a new *FUZZYCONTROL++* project

Opens an existing FUZZYCONTROL++ project and displays the *Open* dialog box, in which you can find and load the required file.

Saves the current project under the current name. If the project has still not been given a name, FUZZYCONTROL++ opens the *Save as...* dialog box.

Opens the *Test File Selection* dialog box

Activates *3-D Graphics*

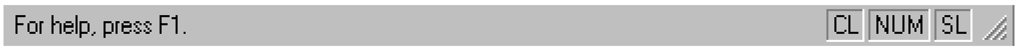
Activates the curve plotter

Opens the *Rule Activity* window

Opens the FUZZYCONTROL++ help window

7.6.2 Status bar

The status bar provides you with information about the current operating status. If and is displayed on the left and right corners as you work with FUZZYCONTROL++, if there is a checkmark in front of *Status Bar* in the *View* menu. The status bar is displayed on the lower edge of the FUZZYCONTROL++ working window.


For help, press F1. CL NUM SL

Left-hand side of the status bar

As you move through the menus with the mouse or the cursor keys, this part displays a short description of the menu command selected or the button on the toolbar selected with the mouse pointer.

Right-hand side of the status bar

This displays which of the following keys are " activated ":

- *CL* CAPS LOCK is activated.
- *NUM* NUM LOCK (right-hand keypad for numeric input) is activated.
- *SL* The SCROLL LOCK key is activated.

Note: You receive assistance while using *FUZZYCONTROL++* through a yellow information box that appears after a short time if you hold the mouse pointer over the same position.

7.7 Menu: Window

With the *Window* menu you can determine how the windows of *FUZZYCONTROL++* are displayed. It contains the following commands:

- ***Cascade***

  All windows except those that have been minimized to icon size are positioned overlapping diagonally and all have the same size. The active window is in the foreground.

- ***Tile***

  All windows except those that have been minimized to icon size are arranged next to each other and fill the space (if there is more than one).

- ***Arrange Icons***

  The windows that have been minimized to icon size are arranged along the lower edge of the working window of *FUZZYCONTROL++*. If there is an opened project window there, some or all of the icons may be covered.

- ***Window1, 2,...***

  At the end of the *Window* menu there is a list of opened windows. A checkmark appears in front of the name of the active window. If you select an entry in this list, you can switch directly to that window, which is then the active window.

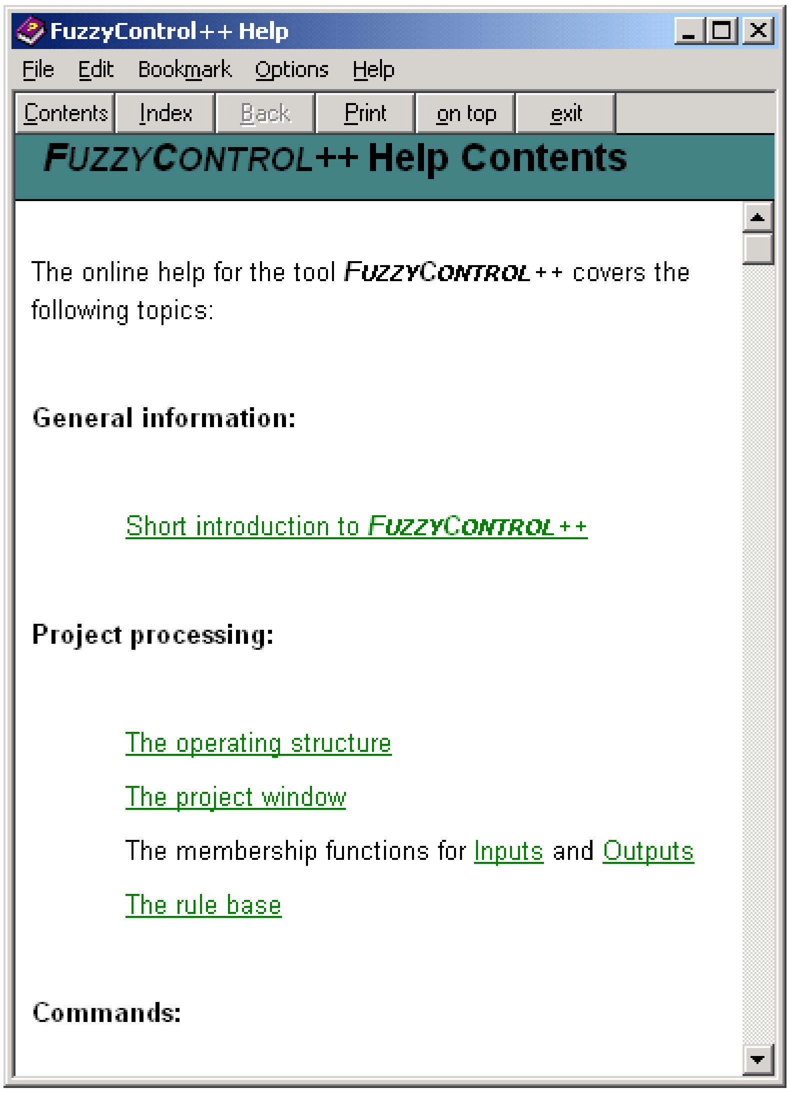
Note: Use *Window1, Window2, ...* in the *Window* menu to get an overview of the windows that are open because some of them might be minimized or covered by other windows.

7.8 Menu: Help (“?”)

The *Help(“?”)* menu provides you with assistance when using *FUZZYCONTROL++*. It is subdivided into the following sections:

- **Index**

This part gives you an overview of topics about which help is available. From this window, you can obtain information by going straight to a specific topic or by opening the subcategories until you reach the desired topic.


FuzzyControl++ Help
File Edit Bookmark Options Help
Contents Index Back Print on top exit

FUZZYCONTROL++ Help Contents

The online help for the tool FUZZYCONTROL++ covers the following topics:

General information:

Short introduction to FUZZYCONTROL++

Project processing:

The operating structure

The project window

The membership functions for Inputs and Outputs

The rule base

Commands:

By clicking on the words highlighted in color and underlined in the help text you can jump to further help topics. To return to the previous information, click on *Back* in the header bar of each help window. The *Contents* calls up the first help window directly. Clicking on the green words with broken underlining opens an explanation box.

With the *on top* button, you can keep the help window in the foreground of the screen as you work with *FUZZYCONTROL++*.

- **Using Help**

This command opens the *Help Topics window: Windows help*. Here you can find information about using the Windows help system.

- **About *FUZZYCONTROL++***

Here you can obtain information about your copy of the *FUZZYCONTROL++* program and contact information.

Use this command to display the copyright message and the *FUZZYCONTROL++* version number.

Appendix

A.1 Example: Control of an awning (window blinds system)

Creation of a nonlinear fuzzy system with *FuzzyControl++*

This example is intended to help you get started with *FuzzyControl++*. It explains step by step how you can use *FuzzyControl++* to generate a fuzzy system with linguistically defined input/output characteristics. Emphasis is placed on explanation in this example: The aim is to model an "intelligent" control of an awning that depends on the light available and the time of day.

A.1.1 Create a new project

Create a new project with the *File/New* command (or by clicking on the relevant icon on the toolbar). The *Define Project* window opens and prompts you to enter the number of inputs and outputs. Enter **two inputs** and **two outputs**. Click on *OK* and a window appears with a block for the fuzzy system ("if ... then") and the blocks for the two inputs and the two outputs. Save the project under the name "Awning" (*File/Save as...* menu command).

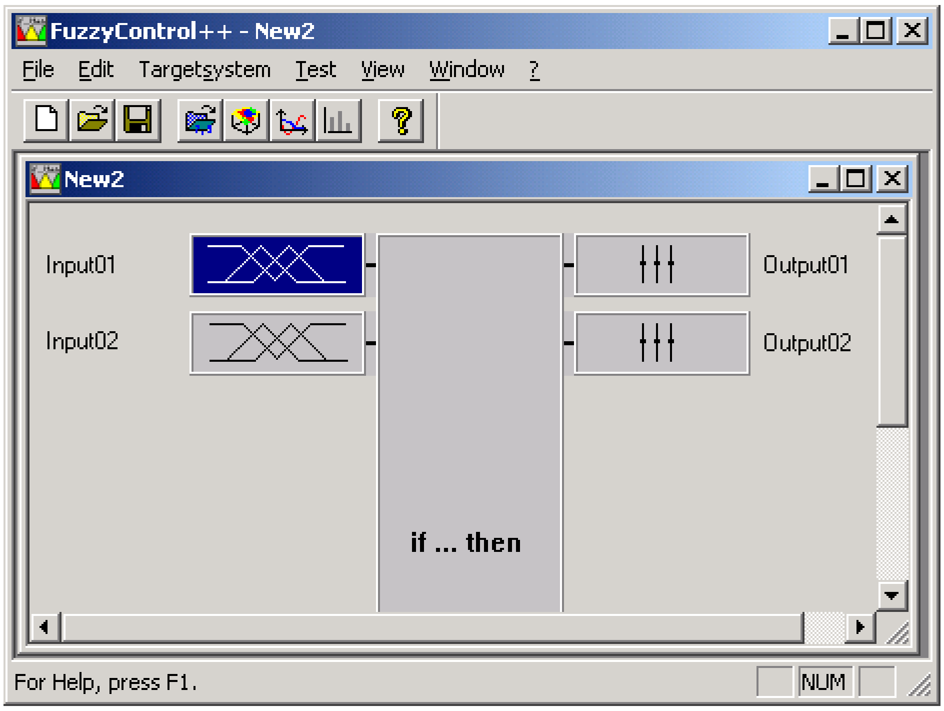


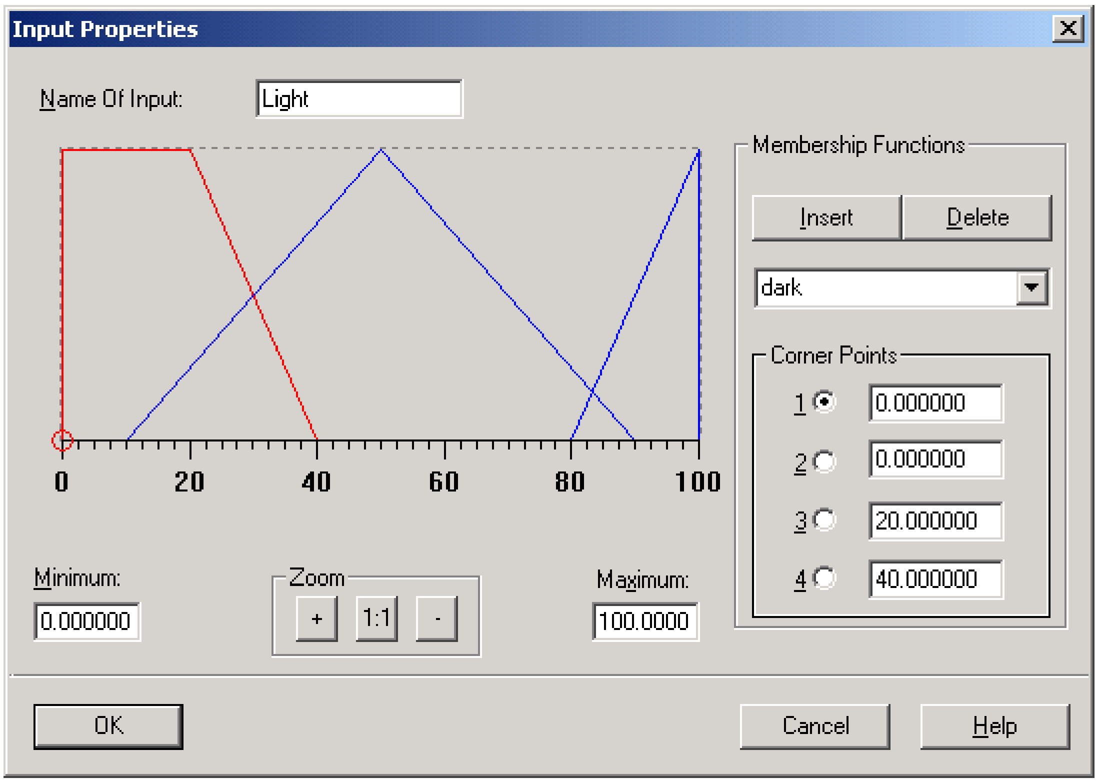
Fig. 1: New fuzzy project with two inputs and two outputs

A.1.2 Edit the inputs and outputs

Click on the "**Input01**" block with the right mouse button (alternatively: double-click with the left mouse button) and select the *Input Properties....* option, which opens the *Input Properties* window. Enter "**Light**" as the name of this input. To assign different linguistic values to the linguistic variables "Light" and to assign these in turn to certain membership functions, click on the *Insert* button in the membership function section. Now you are prompted to enter the number of linguistic values (and therefore the number of membership functions) you want to assign to the "Light" variable. It makes sense to subdivide the light conditions into **three** ranges: **dark**, **ideal** and **bright**. Change the name of the linguistic values "negative", "zero" and "positive" in the selection field, which appear automatically.

You must now assign the shapes of the membership functions to these three values. At this point, human "fuzzy knowledge" is included: Nobody is going to be able to say precisely where to draw a "sharp" boundary between "dark" and "ideal". You are more likely to encounter statements like: "The light conditions are fairly good but it is maybe a bit darker than would be ideal." A "light expert" such as a photographer may be able to make other (possibly more precise) statements about the "light conditions" than a layperson because of his professional training and experience.

Estimates of this kind would lead to the following design: The light condition "pitch dark" is assigned **0** ("**Minimum**" value) and "fully bright" is assigned **100** ("**Maximum**" value). 50 is estimated as "ideal". Between these values there are transition zones where you can make statements like "Now it is x% ideal and y% dark". With *FUZZYCONTROL++* you can create trapezoidal membership functions in general (described by four corner points), including rectangular functions, triangular functions and singletons, which are special cases of trapezoids. Select the three membership functions as shown in Fig. 2. Note that you can only ever edit the function selected in the selection box (shown in red). It is only possible to select it with this selection box and not by clicking a curve. You can edit the function shown in red using the corner point entry fields in the "corner points" section. However, it is also possible to move the corner points of the red function directly by clicking and dragging them with the mouse. If a corner point is marked, it is indicated by a red circle around the point.


Input Properties

Name Of Input: Light

Membership Functions: dark

Corner Points:
1 (●) 0.000000
2 ( ) 0.000000
3 ( ) 20.000000
4 ( ) 40.000000

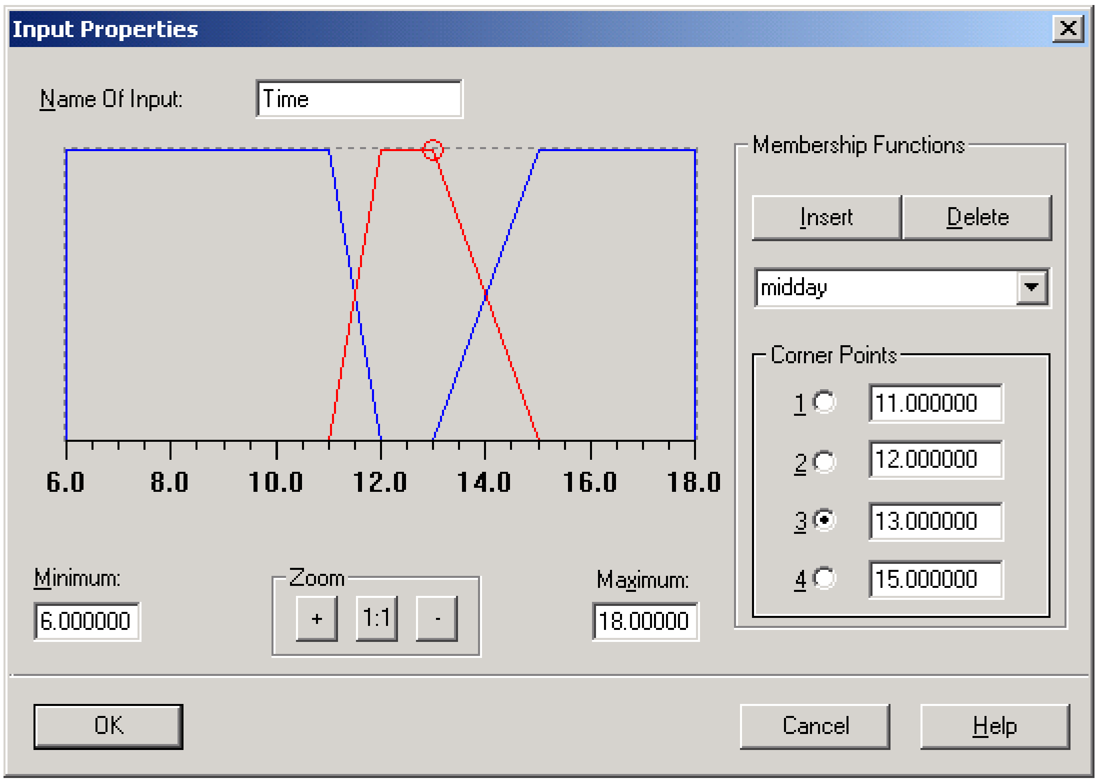
Minimum: 0.000000

Maximum: 100.0000

Fig. 2: Editing the first input "Light"

Confirm your entries with *OK*.

Now click with the right mouse button on the "**Input02**" block (alternatively: double-click with the left mouse button) and edit the second input. Give it the name "**Time**". Because you only want to use the awning during the day, you only need to consider the time between 6 a.m. and 6 p.m. So enter 6 and 18 (using the 24 hour clock) as the "Minimum" and "Maximum" values and click on the "Insert" button to define the membership functions. Once again **three** linguistic values would seem obvious: **morning**, **midday** and **afternoon**. The transitions between the times of day are continuous, which the fuzzy principle is appropriate for. The "main midday time" is the time between 12:00 and 13:00; but between 11:00 and 12:00, and between 13:00 and 15:00 we have fuzzy sets. These estimations are reflected in the three membership functions selected (see Fig. 3).



Name Of Input: Time

midday

1 11.000000

2 12.000000

3 13.000000

4 15.000000

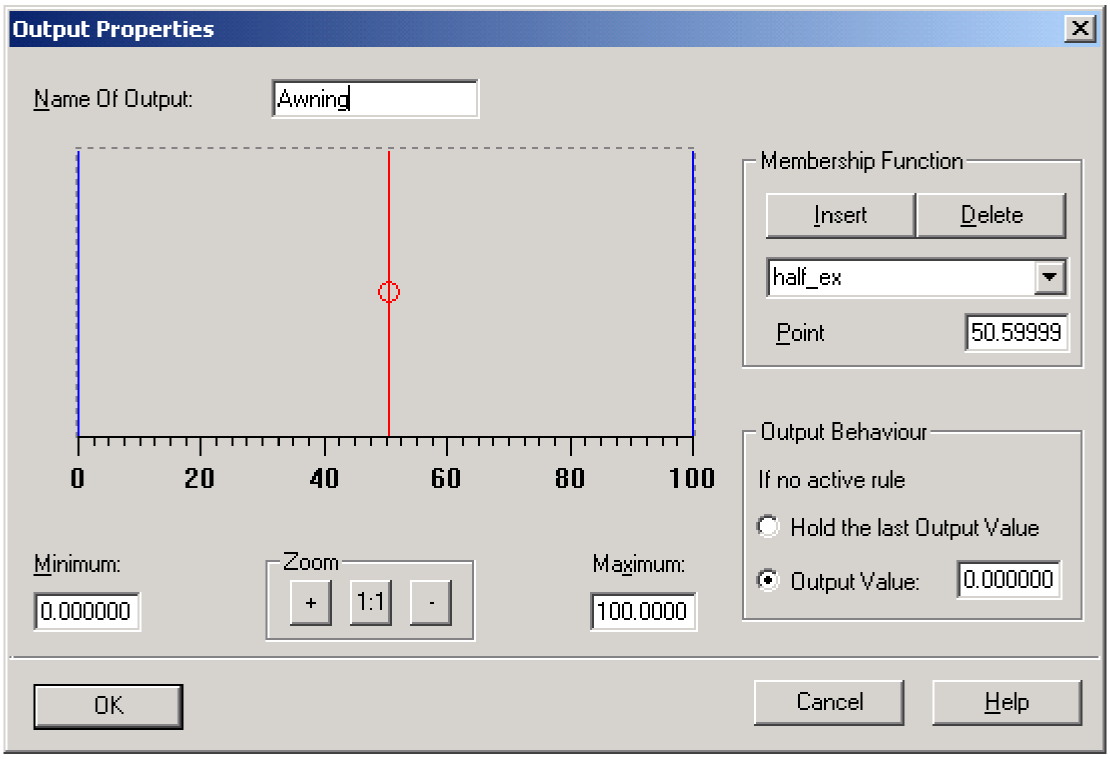
Minimum: 6.000000

Maximum: 18.00000

Fig. 3: Editing the second input "Time"

Confirm your entries with *OK*.

Now edit the "**Output01**" block in the same way. Give it the name "**Awning**", select **0** and **100** as the lower and upper limits and insert three membership functions: **retracted**, **half extended** and **extended**. Unlike the inputs, only singletons are permitted as the outputs. Fig. 4 shows the selection of singleton positions that are most suitable for us.



Name Of Output: Awning

Membership Function: half_ex

Point: 50.59999

Output Behaviour

If no active rule

Hold the last Output Value [✓ Output Value:] 0.000000

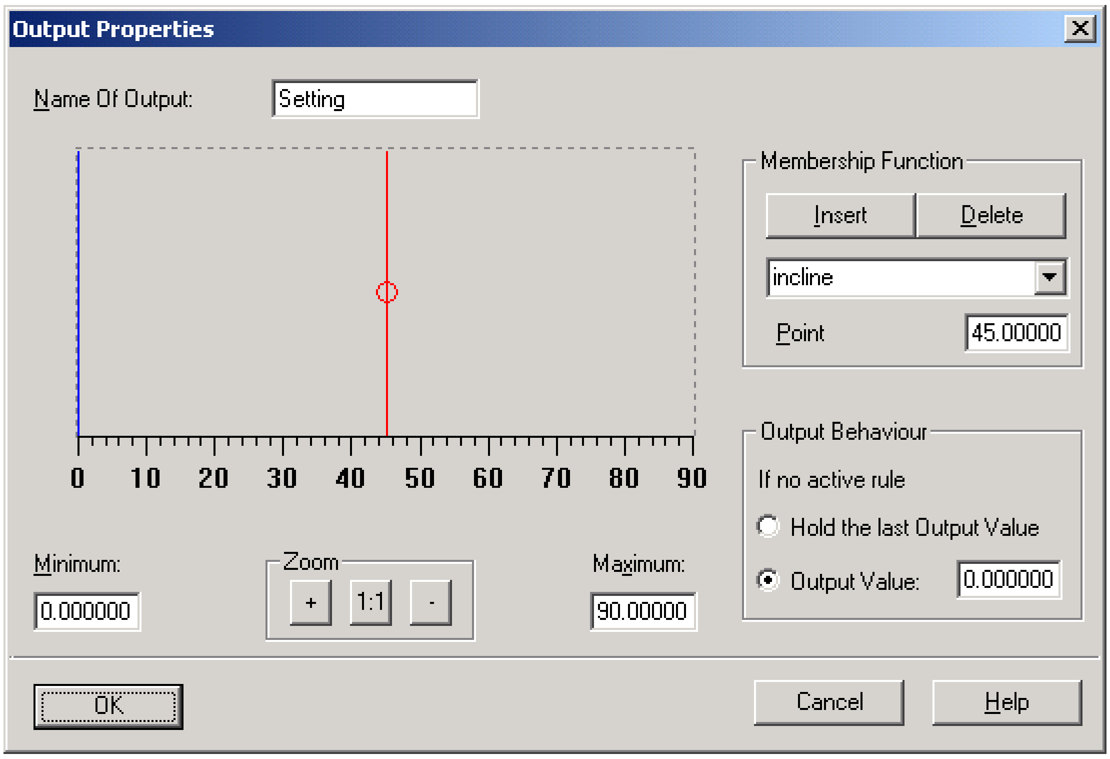
Minimum: 0.000000

Maximum: 100.0000

Fig. 4: Editing the first output "Awning"

The second output might be the setting of the angle to the horizontal. Therefore edit the "**Output02**" block, give it the name "**Setting**", select **0** (meaning 0 degrees, horizontal position) and **90** (meaning 90 degrees, vertical position) as the limits of the range, insert the **second** membership functions **straight** and **inclined** and set the singletons to positions 0 (straight) and 40 (inclined).

Confirm the output settings with *OK*.


Output Properties

Name Of Output: Setting

Membership Function: incline

Point: 45.00000

Output Behaviour

If no active rule

( ) Hold the last Output Value

(●) Output Value: 0.000000

Minimum: 0.000000

Maximum: 90.00000

Fig. 5: Editing the second output "Setting"

A.1.3 Set up the fuzzy rules

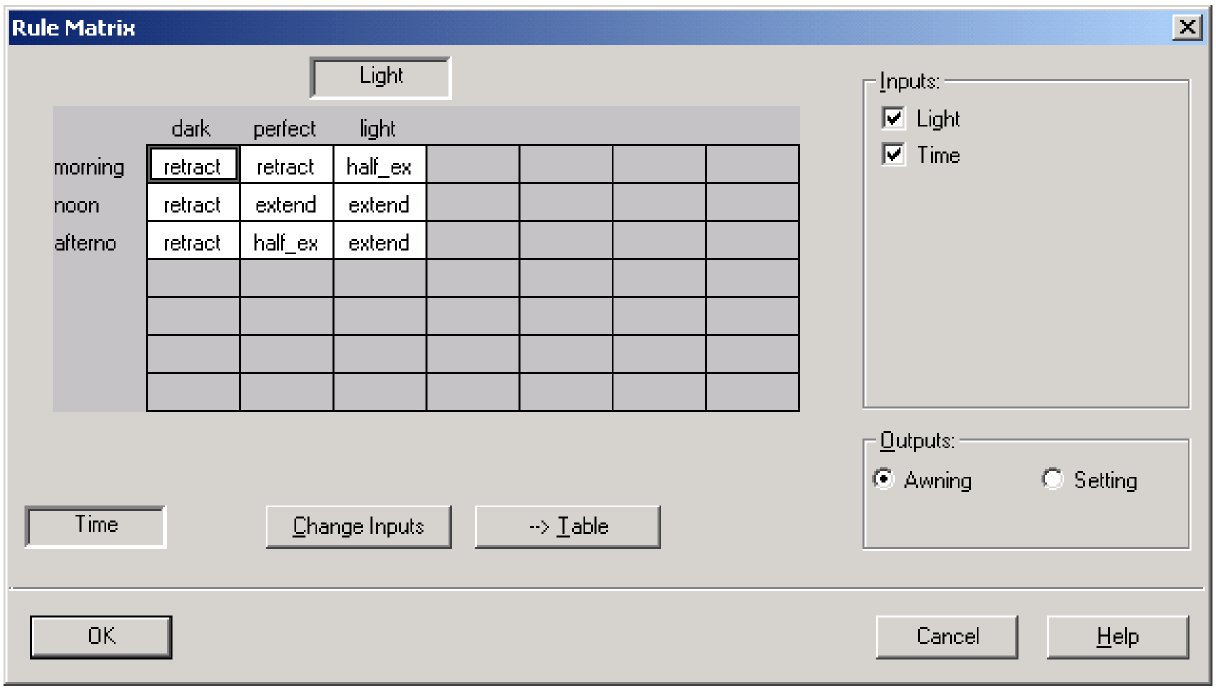
FuzzyControl++ provides you with two ways of writing the "if...then" rules: The rule table and the rule matrix. It makes sense to generate a complete set of rules using the matrix form, recommended for this example, which is a more compact way of viewing the complete set of rules. If the both 3 x 3 rule matrices (one for each output) are fully assigned, you have a complete set of rules.

Click on the *if...then* fuzzy block with the right mouse button and select the *Matrix...* entry (alternatively: *Edit/Rules.../→Matrix*). The *Rule matrix* window opens. The linguistic variables of the inputs (here: light and time) and the associated linguistic values (here: dark, ideal, bright or morning, midday, afternoon) are displayed in the form of a matrix. Each of the nine lightly colored (i.e. editable) fields stands for an IF part of a rule (for example: if light=dark and time =midday...). Now you must associate these IF parts with the suitable THEN part for each output by clicking the matrix element in question and selecting the appropriate THEN part.

As an example, let us look at the element in the top left of the rule matrix for the "Awning" output: A sensible rule is:

If light = dark **and** time = morning, **then** awning = retracted.

Fill in the matrix for the "Awning" output as shown in Fig. 6.



	dark	perfect	light
morning	retract	retract	half_ex
noon	retract	extend	extend
afterno	retract	half_ex	extend

Inputs: [✓] Light [✓] Time

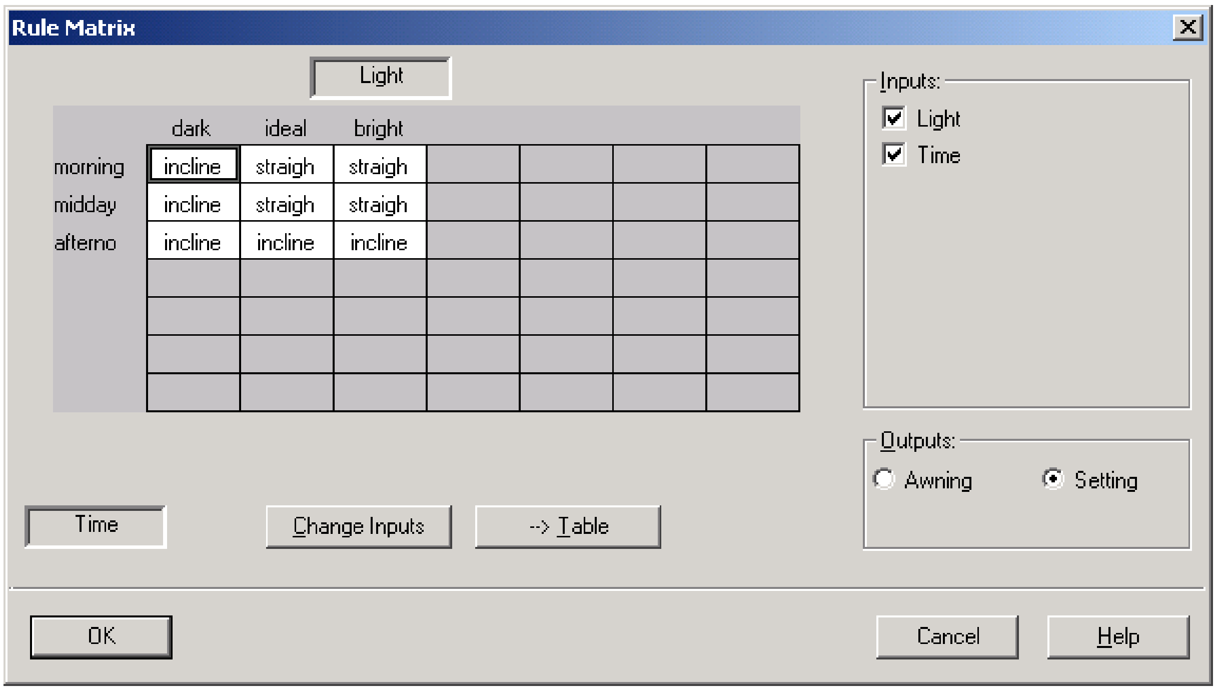
Outputs: [✓ Awning] Setting

Fig. 6: Rule matrix for the first output "Awning"

When writing the rules you can, for example, take account of the fact that with the same light conditions it is usually warmer in the afternoon than in the morning, so you must extend the awning further in the afternoon than in the morning.

The rule matrix for the "Setting" output is entered in the same way. For example, it takes account of the fact that the awning must be inclined if it is relatively dark because then the water can run off better if it rains. We also want the awning to be a little more inclined during the afternoon because it is fitted on the west side of a building and the sun comes in at an increased angle during the afternoon.

So fill in the matrix for the "Setting" output as follows:


Rule Matrix

Light

	dark	ideal	bright
morning	incline	straigh	straigh
midday	incline	straigh	straigh
afterno	incline	incline	incline

Inputs: [✓] Light [✓] Time

Outputs: ○ Awning ● Setting

Fig. 7: Rule matrix for the second output "Setting"

The fuzzy system is now fully parameterized.

A.1.4 Visualization of the system characteristic

The *Test* menu provides various tools for viewing the input/output characteristic and checking the parameters chosen (i.e. the membership functions and rules). The *3-D Representation* option is an easy way of visualizing the system behavior.

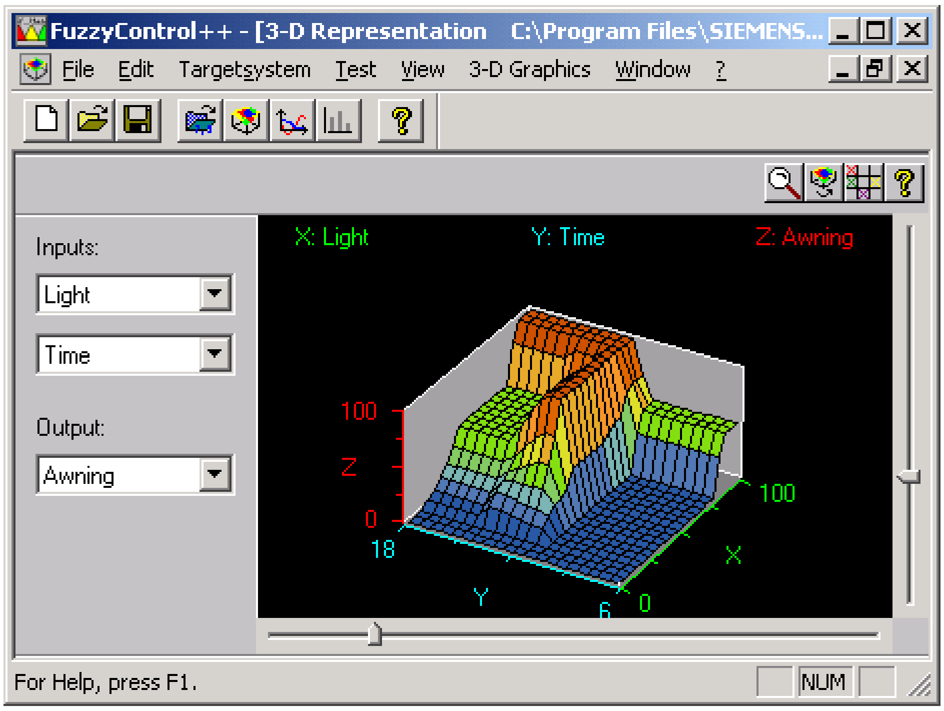

FuzzyControl++ - [3-D Representation C:\Program Files\SIEMENS...
File Edit Targetsystem Test View 3-D Graphics Window ?
Inputs:
Light
Time
Output:
Awning
X: Light
Y: Time
Z: Awning
100
Z
0
18
Y
6
0
X
100
For Help, press F1.
NUM

Fig. 8: Visualization via 3-D Representation: "Awning" output

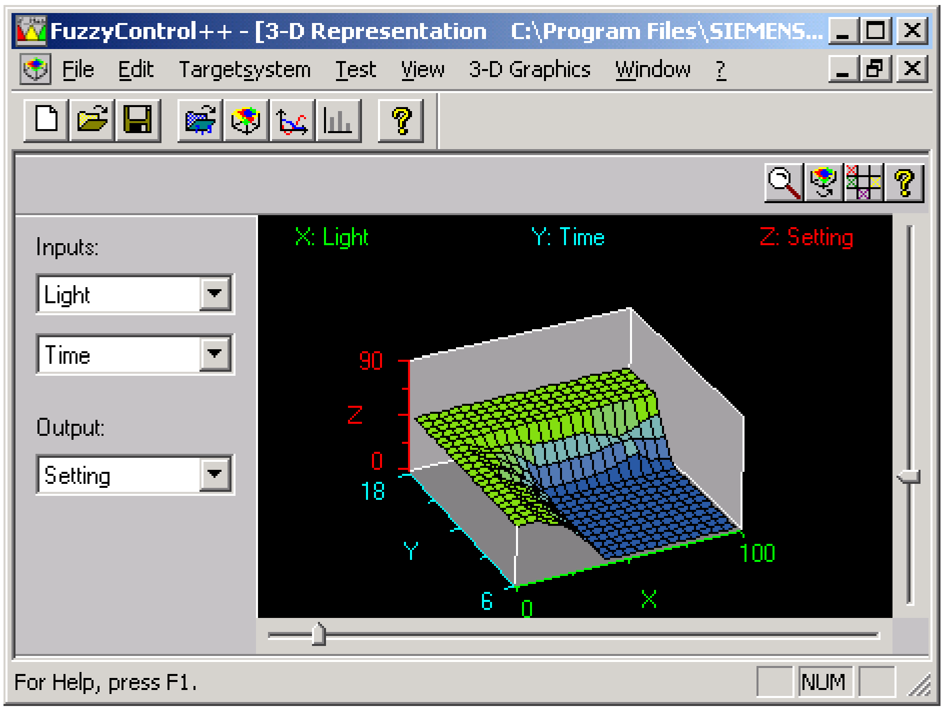

FuzzyControl++ - [3-D Representation C:\Program Files\SIEMENS...
File Edit Targetsystem Test View 3-D Graphics Window ?
Inputs:
Light
Time
Output:
Setting
X: Light
Y: Time
Z: Setting
90
Z
0
18
Y
6
0
X
100
For Help, press F1.
NUM

Fig. 9: Visualization via 3-D Representation: "Setting" output

If you are still not happy with the input/output behavior, you can optimize it further by modifying the rules and/or membership functions (number and shape). You can also insert new inputs and outputs subsequently to take account of other influencing variables or create further output signals. In our example an additional input called "Season" would be a possibility.

A further visualization option is the curve plotter. To open the curve plotter, choose the *Curve Plotter* option in the *Test* menu. In offline operation you can click the curve generator button to assign each of the inputs a constant or configurable (three point) signal. Recording starts when you click ▶.

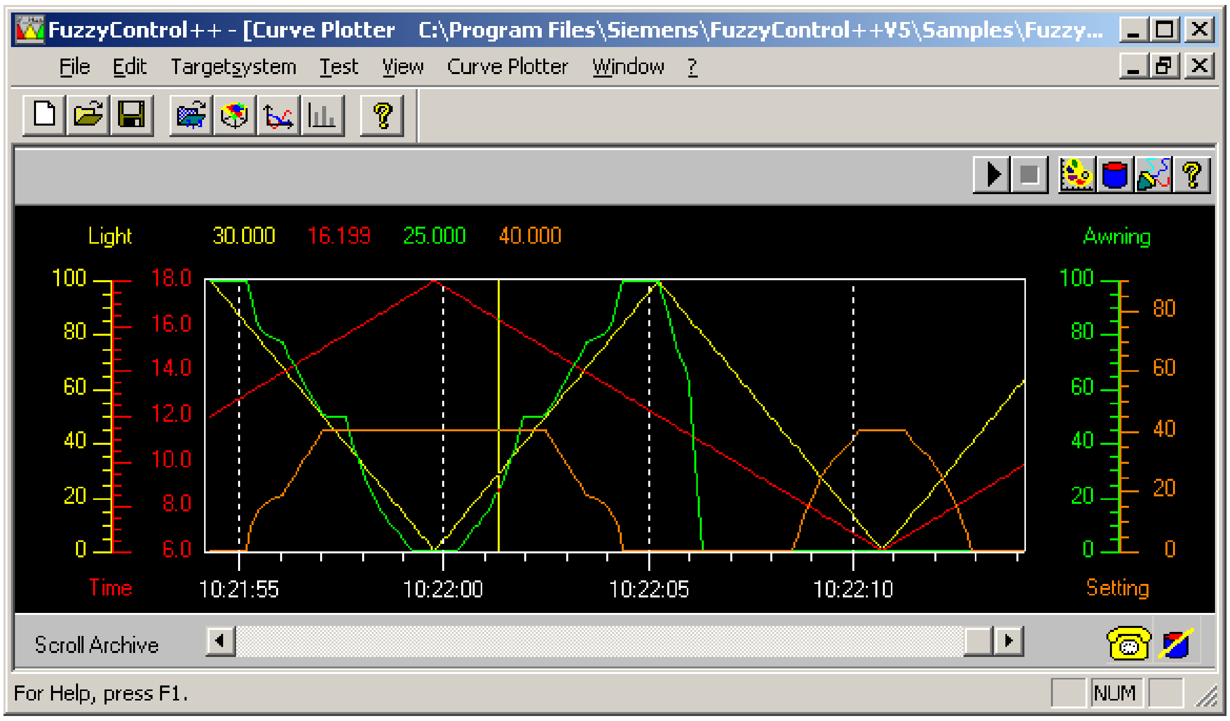

FuzzyControl++ - [Curve Plotter C:\Program Files\Siemens\FuzzyControl++V5\Samples\Fuzzy...
File Edit Targetsystem Test View Curve Plotter Window ?
Light 30.000 16.199 25.000 40.000 Awning
100 80 60 40 20 0
18.0 16.0 14.0 12.0 10.0 8.0 6.0
100 80 60 40 20 0
80 60 40 20 0
Time 10:21:55 10:22:00 10:22:05 10:22:10 Setting
Scroll Archive
For Help, press F1.
NUM

Fig. 10: Visualization with the curve plotter

The current values of the four signals recorded are displayed continuously in the color assigned to the curve. The time axis is automatically labeled with the current time. After you have stopped the curve plotter by clicking on ■, a diagram appears with a vertical read-off line. You can click this with the mouse and drag it over the entire visible range of the diagram. The curve values for the position of the read-off line are displayed.

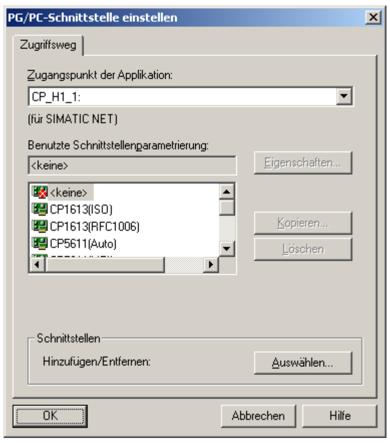
A.1.5 Saving the project

After you have configured and tested the fuzzy system, you can save the project as an *.fpl file (e.g. "awning.fpl") with *File/Save* or *File/Save as....*

A.2 Example Configuration

A.2.1 Creating an example configuration for an MPI connection

- Firstly, open the PG/PC interface settings in the Control Panel.


PG/PC-Schnittstelle einstellen

Zugriffsweg

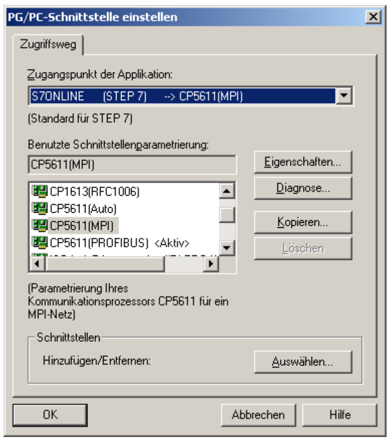
Zugangspunkt der Applikation: CP_H1_1:
(für SIMATIC NET)

Benutzte Schnittstellenparametrierung: <keine>

<keine>
CP1613(ISO)
CP1613(RFC1006)
CP5611(Auto)

Schnittstellen
Hinzufügen/Entfernen:

- Most access points are not assigned to a component. Therefore, choose the desired access point from the upper list e.g. S7ONLINE and the relevant component from the lower list e.g. CP5611 (MPI). After this, the assignment is shown in the form of an arrow in the upper field.


PG/PC-Schnittstelle einstellen

Zugriffsweg

Zugangspunkt der Applikation:
S7ONLINE (STEP 7) --> CP5611(MPI)
(Standard für STEP 7)

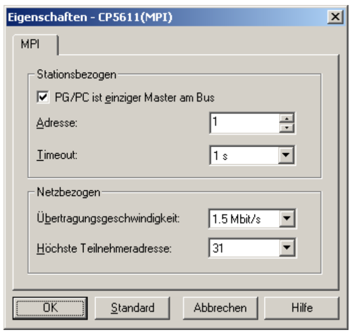
Benutzte Schnittstellenparametrierung:
CP5611(MPI)

CP1613(RFC1006)
CP5611(Auto)
CP5611(MPI)
CP5611(PROFIBUS) <Aktiv>

(Parametrierung Ihres Kommunikationsprozessors CP5611 für ein MPI-Netz)

Schnittstellen
Hinzufügen/Entfernen:

- The access point must now be appropriately configured. To do this select the access point to be parameterized now from the upper list, S7ONLINE -> CP5611 in this example, and click on the properties button. In the following window, make sure that the PG/PC check box is selected as the only master on the bus. Select and make a note of the transfer rate e.g. 1.5 Mbit/s. This must correspond with the settings that follow later.


Eigenschaften - CP5611(MPI)

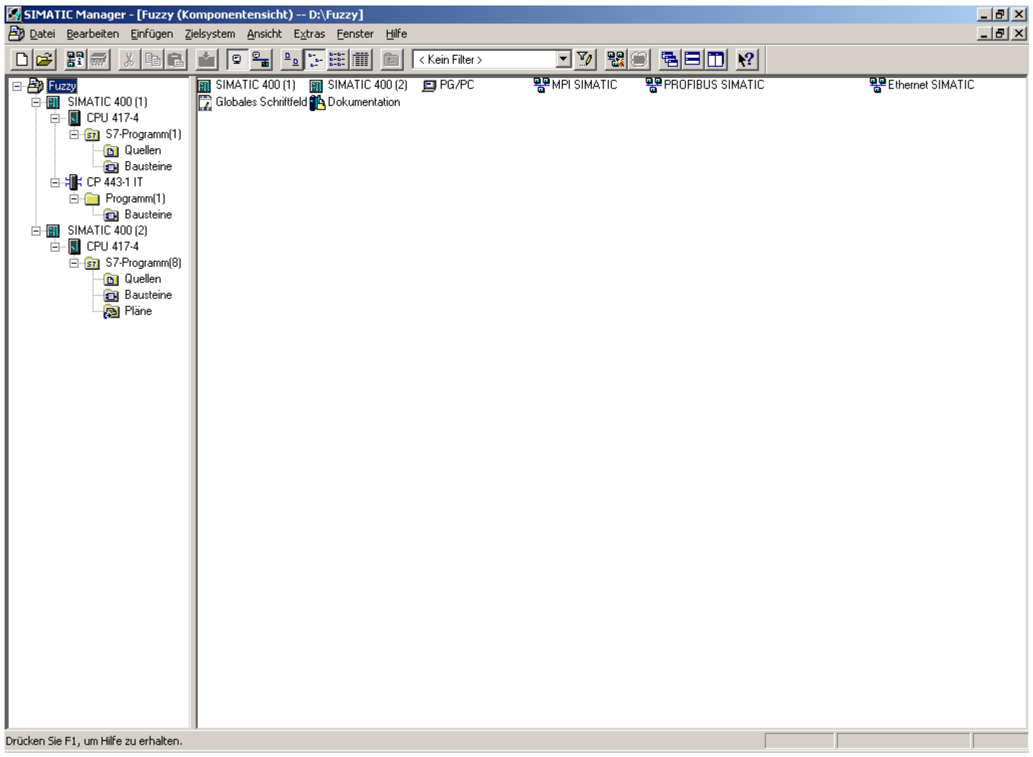
MPI

Stationsbezogen
[✓] PG/PC ist einziger Master am Bus
Adresse: 1
Timeout: 1 s

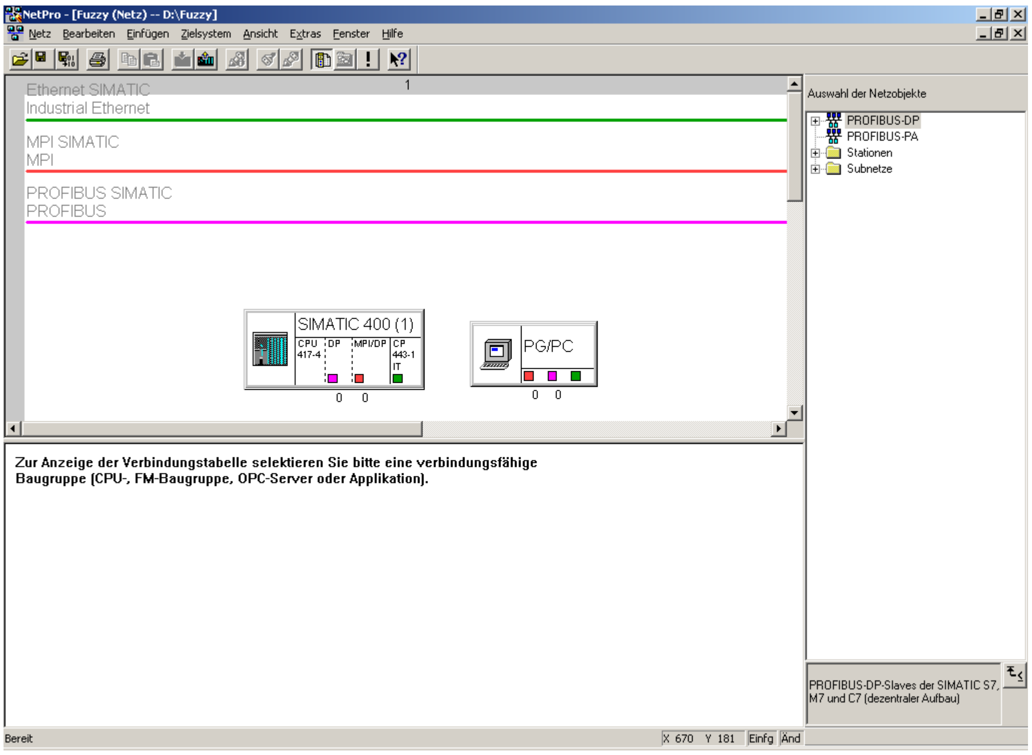
Netzbezogen
Übertragungsgeschwindigkeit: 1.5 Mbit/s
Höchste Teilnehmeradresse: 31

- Now close the properties window and then the PG/PC-interface settings window by clicking on OK and confirm your settings in the following popup dialog box.

- You must now create an offline network configuration and load it into the SIMATIC. To do this, open the SIMATIC Manager and your project. In this example, the project is called D:\Fuzzy.

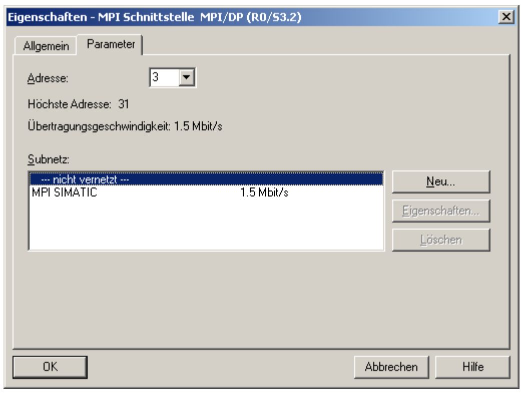


- Select a SIMATIC CPU and double-click on connections in the right-hand window. Upon this, the NetPro program opens with the current configuration.



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

- To reallocate an MPI address to the SIMATIC, double-click on the MPI-Interface (red box). Enter the desired address e.g. 3. In the lower Subnets window you can see, that in the current example that the interface is not yet networked. Therefore, click on the desired network, MPI SIMATIC in this case. If the transfer rate displayed e.g. 1.5 Mbit/s, doesn't correspond to that selected in the PG/PC interface settings, this can be adjusted through properties and network settings.



Adresse: 3

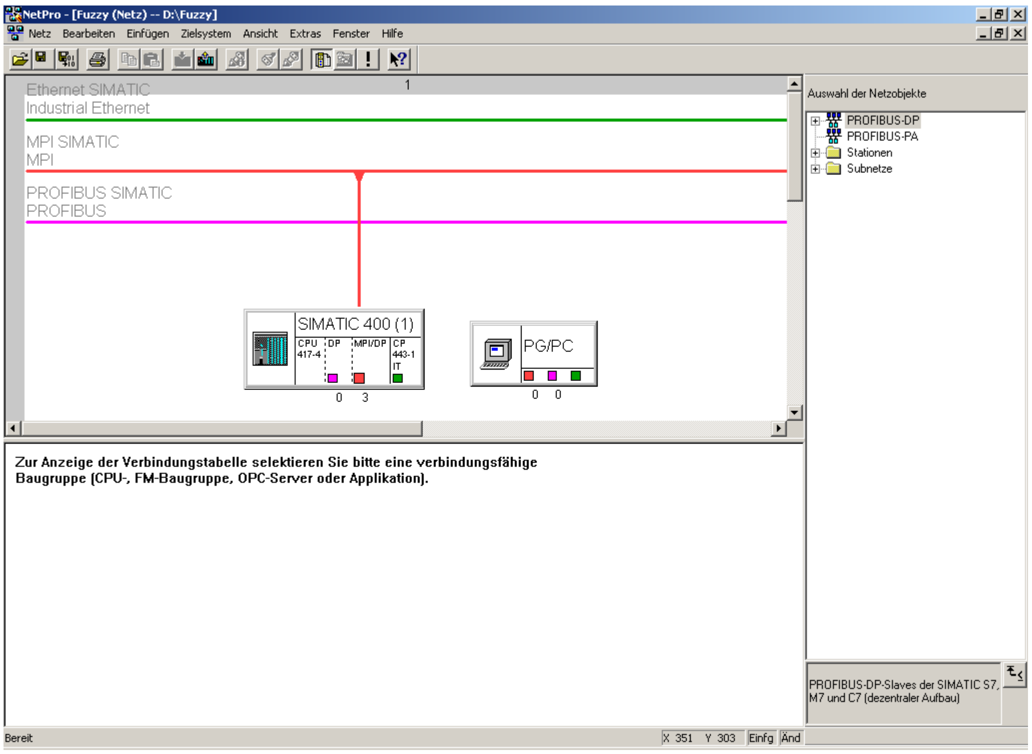
Höchste Adresse: 31

Übertragungsgeschwindigkeit: 1.5 Mbit/s

Subnetz:

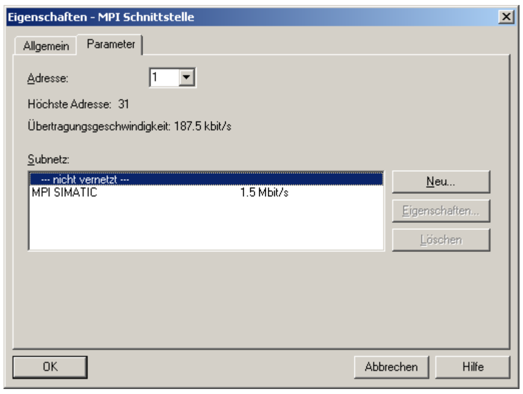
--- nicht vernetzt ---	
MPI SIMATIC	1.5 Mbit/s

- Confirm your settings with OK. A connection between the SIMATIC and the MPI Bus is now displayed (red line).



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

- The PG/PC are put on the bus in a similar manner. Double-click the MPI interface (red box) and enter the desired address e.g. 1. Of course, the address of the MPI interface must be different to that of the SIMATIC. Create the desired connection, by clicking on the corresponding Subnet, MPI SIMATIC in this case. Make sure that the selected transfer rates correspond with each other and change them if necessary, as described above.



Adresse: 1

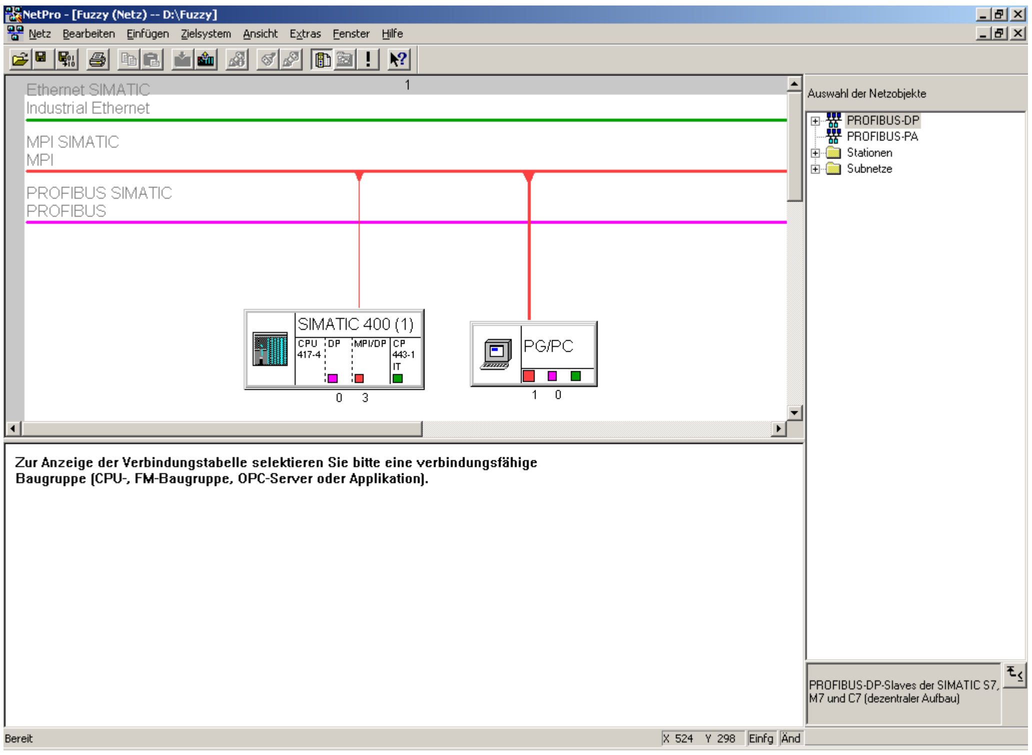
Höchste Adresse: 31

Übertragungsgeschwindigkeit: 187.5 kbit/s

Subnetz:

--- nicht vernetzt ---	
MPI SIMATIC	1.5 Mbit/s

- After confirming your settings with OK, a connection between PG/PC and the MPI bus is displayed (red line).

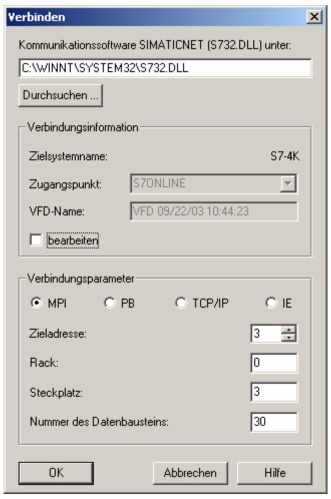


Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

- Lastly you must compile and load into the SIMATIC. To do this, click on ‘save and compile’ and then choose the upload option, after you have selected the desired SIMATIC.

- Now FuzzyControl++ can connect to the SIMATIC via MPI. In the example, you must also select the S7ONLINE access point and the MPI radio button, enter the target address as 3 and click on OK in the FuzzyControl++ Connect dialog box.

- Rack, Slot and data block settings.
  CAUTION: The DB30 data block must be loaded into the S7 and initialized by the Fuzzy function blocks, i.e. it must be run at least once.


Verbinden

Kommunikationssoftware SIMATICNET (S732.DLL) unter:
C:\WINNT\SYSTEM32\S732.DLL

Verbindungsinformation
Zielsystemname: S7-4K
Zugangspunkt: S7ONLINE
VFD-Name: VFD 09/22/03 10:44:23
[ ] bearbeiten

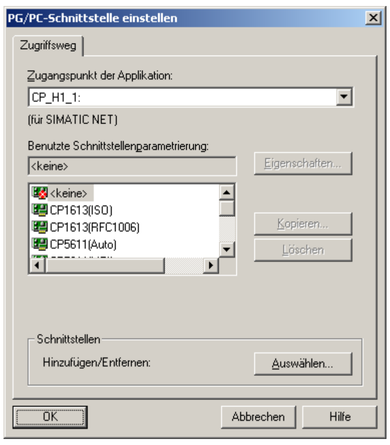
Verbindungsparameter
[✓ MPI] PB TCP/IP IE
Zieladresse: 3
Rack: 0
Steckplatz: 3
Nummer des Datenbausteins: 30

A.2.1.1 *Overview of the configuration for an MPI connection*

PG/PC-interface	Access point: **S7ONLINE** -> CP5611 (MPI)
NetPro	SIMATIC-address: **3** PG/PC-address: 1
FuzzyControl++	Access point: **S7ONLINE** Protocol: **MPI** Target address: **3**

A.2.2 Creating an example configuration for a Profibus connection

- Firstly, open the PG/PC interface settings in the Control Panel.


PG/PC-Schnittstelle einstellen

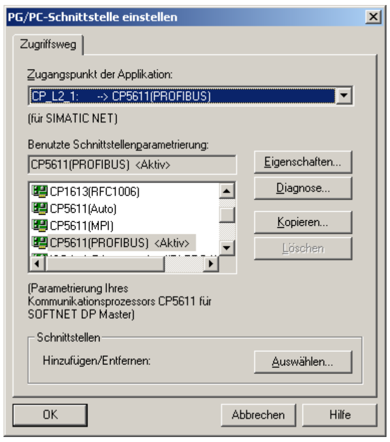
Zugriffsweg

Zugangspunkt der Applikation: CP_H1_1:
(für SIMATIC NET)

Benutzte Schnittstellenparametrierung: <keine>

<keine>
CP1613(ISO)
CP1613(RFC1006)
CP5611(Auto)

- Most access points are not assigned to a component. Therefore, choose the desired access point from the upper list e.g. CP_L2_1 and the relevant component from the lower list e.g. CP5611 (PROFIBUS). After this, the assignment is shown in the form of an arrow in the upper field.



Zugangspunkt der Applikation:
CP_L2_1: --> CP5611(PROFIBUS)
(für SIMATIC NET)

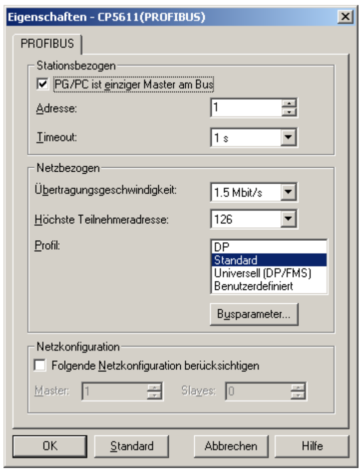
Benutzte Schnittstellenparametrierung:
CP5611(PROFIBUS) <Aktiv>

CP1613(RFC1006)
CP5611(Auto)
CP5611(MPI)
CP5611(PROFIBUS) <Aktiv>

(Parametrierung Ihres Kommunikationsprozessors CP5611 für SOFTNET DP Master)

Schnittstellen
Hinzufügen/Entfernen:

- The access point must now be appropriately configured. To do this select the access point to be parameterized now from the upper list, CP_L2_1 -> CP5611 (PROFIBUS), in this example, and click on the properties button. In the following window, make sure that the PG/PC control check box is selected as the only master on the bus and that the profile is set as standard. Select and make a note of the transfer rate e.g. 1.5 Mbit/s. This must correspond with the settings that follow later.



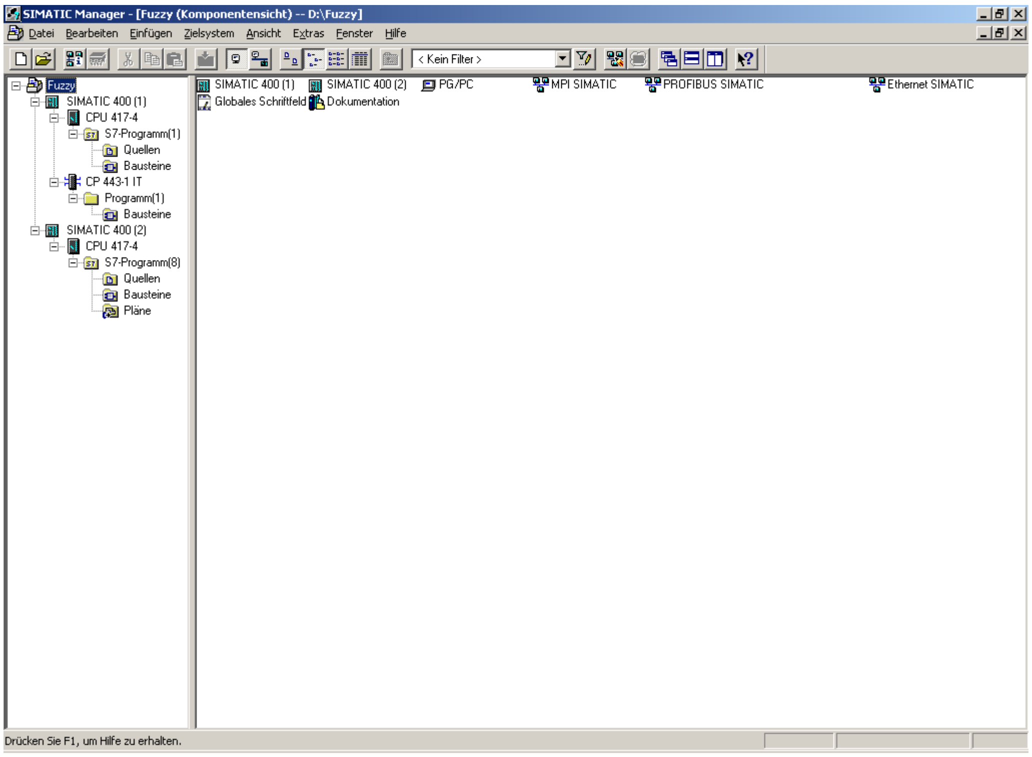
Stationsbezogen
☑ PG/PC ist einziger Master am Bus
Adresse: 1
Timeout: 1 s

Netzbezogen
Übertragungsgeschwindigkeit: 1.5 Mbit/s
Höchste Teilnehmeradresse: 126
Profil: DP / [✓ Standard] / Universell (DP/FMS) / Benutzerdefiniert

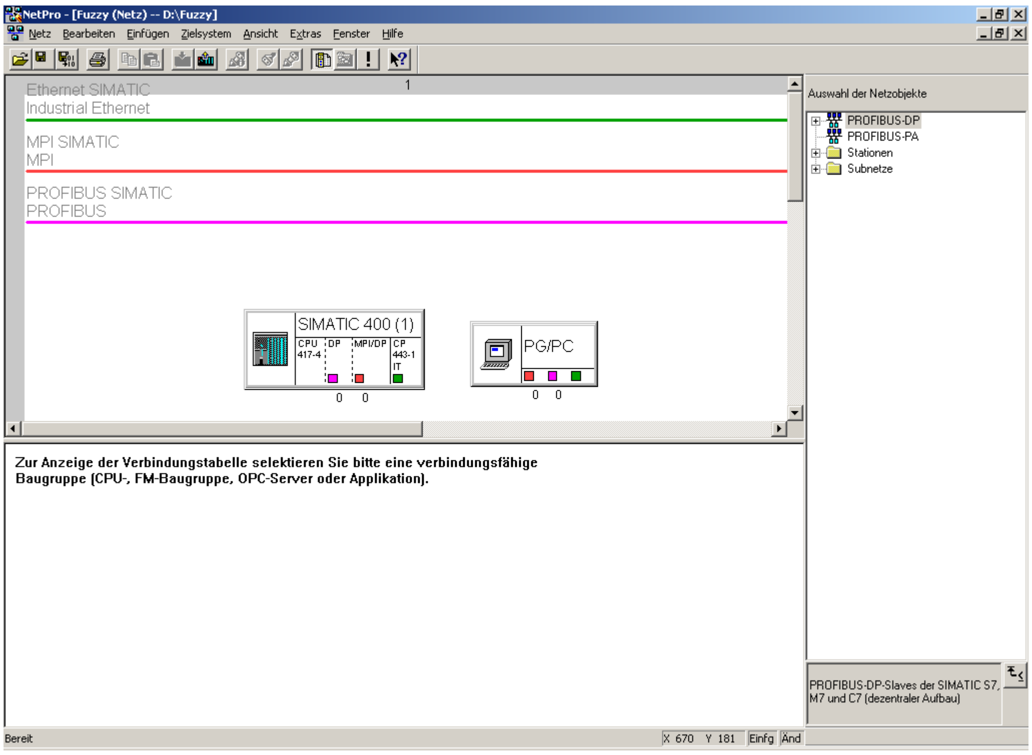
Netzkonfiguration
☐ Folgende Netzkonfiguration berücksichtigen
Master: 1 Slaves: 0

- Now close the properties window and then the PG/PC-interface settings window by clicking on OK and confirm your settings in the following popup dialog box.

- You must now create an offline network configuration and load it into the SIMATIC. To do this, open the SIMATIC Manager and your project. In this example, the project is called D:\Fuzzy.

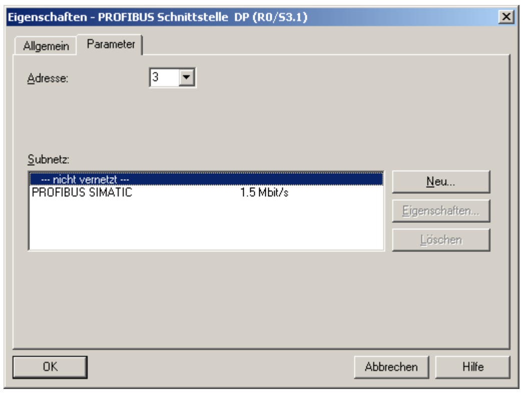


- Select a SIMATIC CPU and double-click on connections in the right-hand window. Upon this, the NetPro program opens with the current configuration.



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

- To reallocate a Profibus address to the SIMATIC, double-click on the Profibus interface (pink box). Enter the desired address e.g. 3. In the lower Subnets window you can see, that in the current example that the interface is not yet networked. Therefore, click on the desired network, PROFIBUS SIMATIC in this case. If the transfer rate displayed e.g. 1.5 Mbit/s, doesn't correspond to that selected in the PG/PC interface settings, this can be adjusted through properties and network settings.

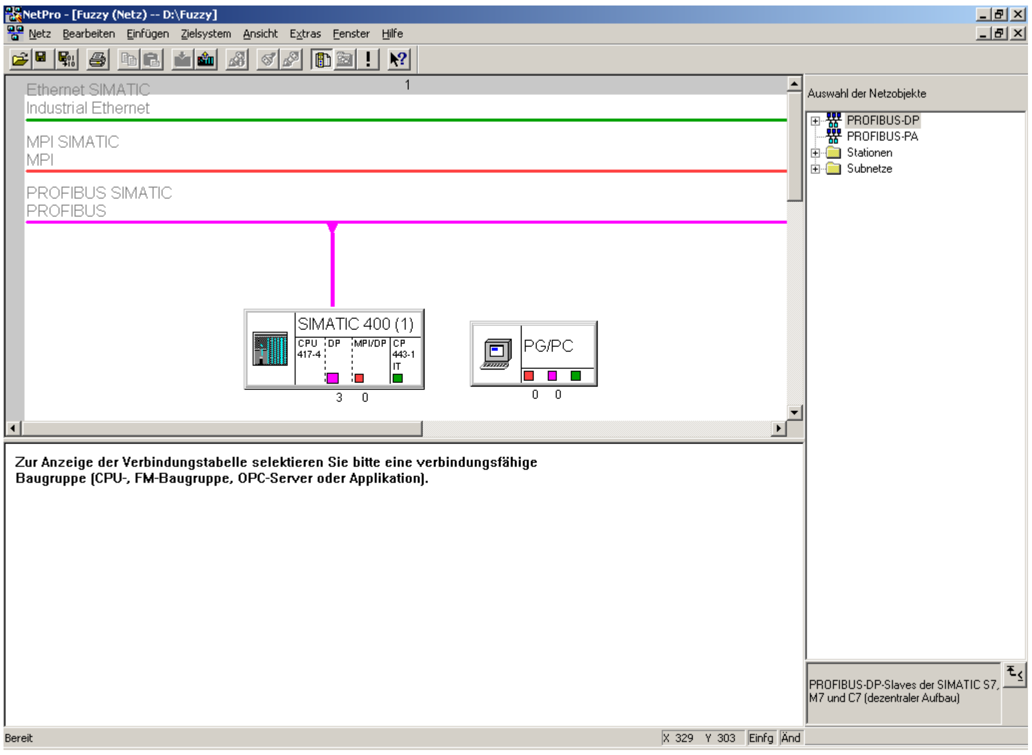


Adresse: 3

Subnetz:

--- nicht vernetzt ---	
PROFIBUS SIMATIC	1.5 Mbit/s

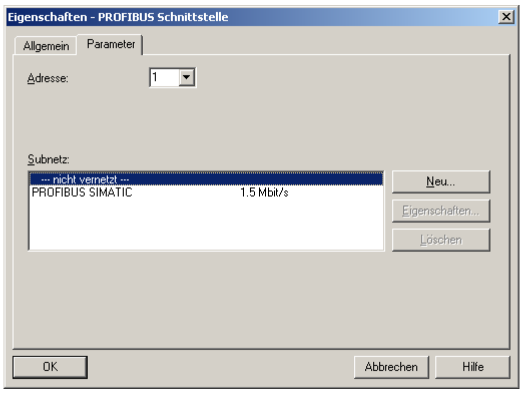
- After confirming your settings with OK, a connection between PG/PC and the Profibus bus is displayed (pink line).



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

- The PG/PC are put on the bus in a similar manner. Double-click the Profibus interface (pink box) and enter the desired address e.g. 1. Of course, the address of the Profibus interface must be different to that of the SIMATIC. Create the desired connection, by clicking on the corresponding Subnet, PROFIBUS SIMATIC in this case. Make sure that the selected transfer rates correspond with each other and change them if necessary, as described above.

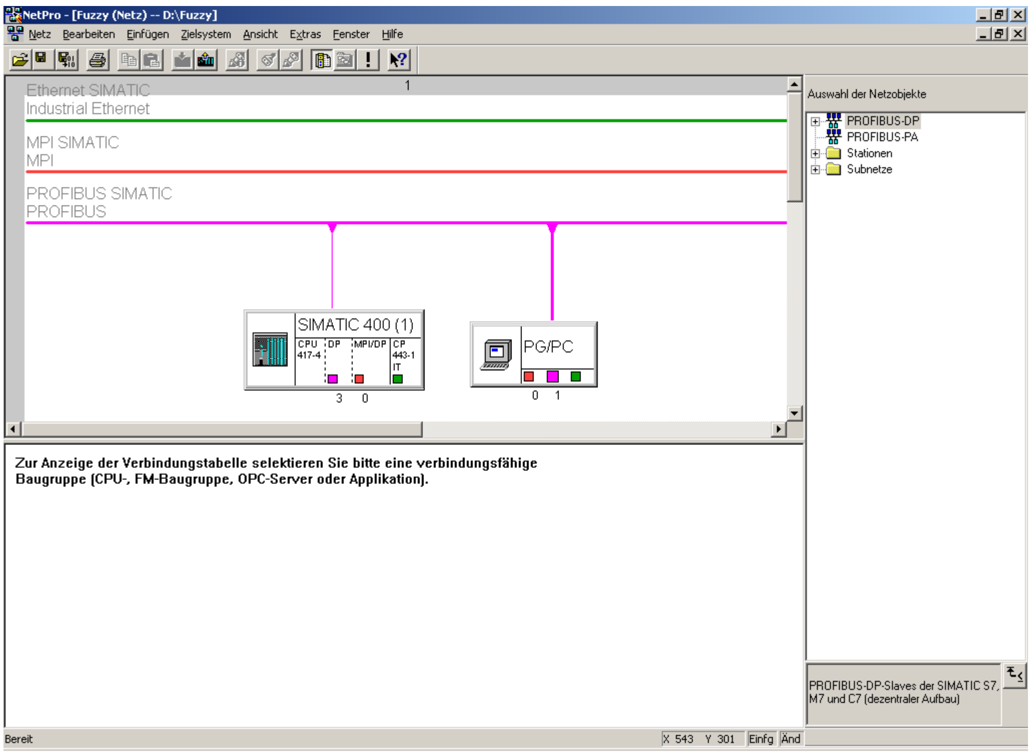


Adresse: 1

Subnetz:

--- nicht vernetzt ---
PROFIBUS SIMATIC 1.5 Mbit/s

- After confirming your settings with OK, a connection between PG/PC and the Profibus bus is displayed (pink line).



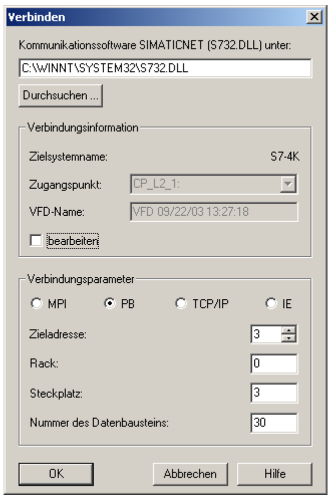
Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

- Lastly you must compile and load into the SIMATIC. To do this, click on 'save and compile' and then choose the upload option, after you have selected the desired SIMATIC.

- Now FuzzyControl++ can connect to the SIMATIC via Profibus. In the example, you must also select the CP_L2_1 access point and the PB radio button, enter the target address as 3 and click on OK in the FuzzyControl++ Connect dialog box.

- Rack, Slot and data block settings.
  CAUTION: The DB30 data block must be loaded into the S7 and initialized by the Fuzzy function blocks, i.e. it must be run at least once.


Verbinden

Kommunikationssoftware SIMATICNET (S732.DLL) unter:
C:\WINNT\SYSTEM32\S732.DLL

Verbindungsinformation
Zielsystemname: S7-4K
Zugangspunkt: CP_L2_1:
VFD-Name: VFD 09/22/03 13:27:18
☐ bearbeiten

Verbindungsparameter
MPI [✓ PB] TCP/IP IE
Zieladresse: 3
Rack: 0
Steckplatz: 3
Nummer des Datenbausteins: 30

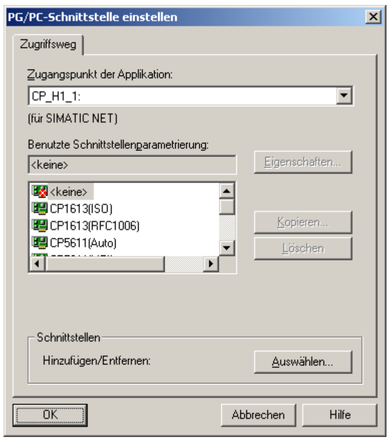
A.2.2.1 Overview of the configuration for a Profibus connection

PG/PC- interface	Access point: **CP_L2_1** -> CP5611 (PROFIBUS)
NetPro	SIMATIC- address: **3** PG/PC- address: 1
FuzzyControl++	Access point: **CP_L2_1** Protocol: **PB**

	Target address: **3**

A.2.3 Creating an example configuration for a TCP/IP connection

- Firstly, open the PG/PC interface settings in the Control Panel.


PG/PC-Schnittstelle einstellen

Zugriffsweg

Zugangspunkt der Applikation:

CP_H1_1:

(für SIMATIC NET)

Benutzte Schnittstellenparametrierung:

<keine>

<keine>
CP1613(ISO)
CP1613(RFC1006)
CP5611(Auto)

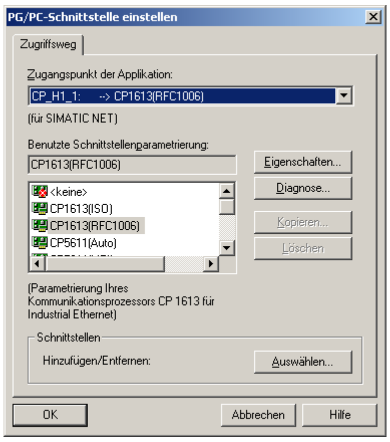
Eigenschaften...
Kopieren...
Löschen

Schnittstellen

Hinzufügen/Entfernen: Auswählen...

OK Abbrechen Hilfe

- Most access points are not assigned to a component. Therefore, choose the desired access point from the upper list e.g. CP_H1_1 and the relevant component from the lower list e.g. CP1613 (RFC1006). After this, the assignment is shown in the form of an arrow in the upper field.


PG/PC-Schnittstelle einstellen

Zugriffsweg

Zugangspunkt der Applikation:
CP_H1_1: --> CP1613(RFC1006)
(für SIMATIC NET)

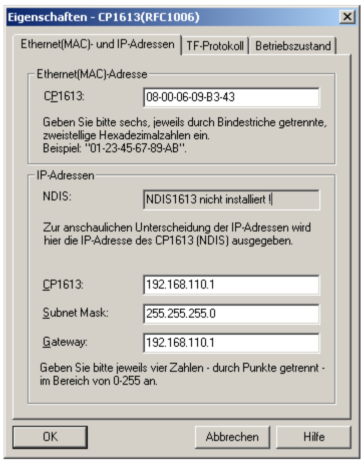
Benutzte Schnittstellenparametrierung:
CP1613(RFC1006)

<keine>
CP1613(ISO)
CP1613(RFC1006)
CP5611(Auto)

(Parametrierung Ihres Kommunikationsprozessors CP 1613 für Industrial Ethernet)

Schnittstellen
Hinzufügen/Entfernen:

- The access point must now be appropriately configured. To do this select the access point to be parameterized now from the upper list, CP_H1_1 -> CP1613 (RFC1006) in this example, and click on the properties button. Enter the desired MAC address of the CP in CP1613 in the Ethernet (MAC) section, e.g. 08-00-06-09-B3-43, and the desired IP address in CP1613 in the IP address section, e.g. 192.168.110.1.


Eigenschaften - CP1613(RFC1006)

Ethernet(MAC)- und IP-Adressen | TF-Protokoll | Betriebszustand

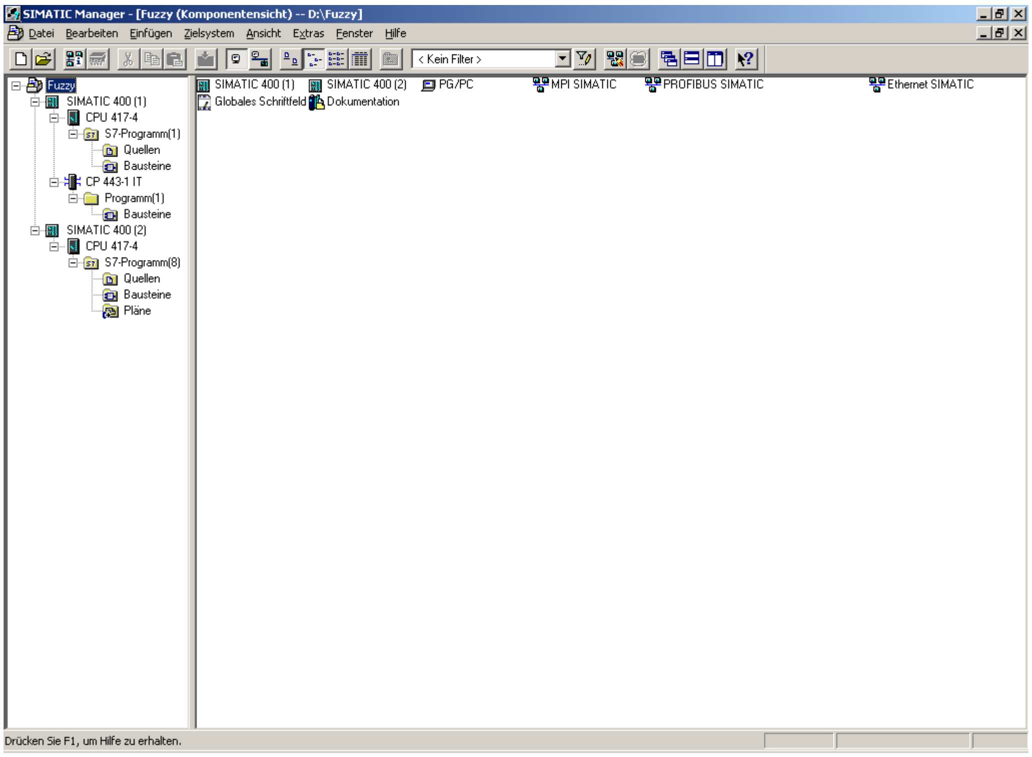
Ethernet(MAC)-Adresse
CP1613: 08-00-06-09-B3-43
Geben Sie bitte sechs, jeweils durch Bindestriche getrennte, zweistellige Hexadezimalzahlen ein.
Beispiel: "01-23-45-67-89-AB".

IP-Adressen
NDIS: NDIS1613 nicht installiert !
Zur anschaulichen Unterscheidung der IP-Adressen wird hier die IP-Adresse des CP1613 (NDIS) ausgegeben.

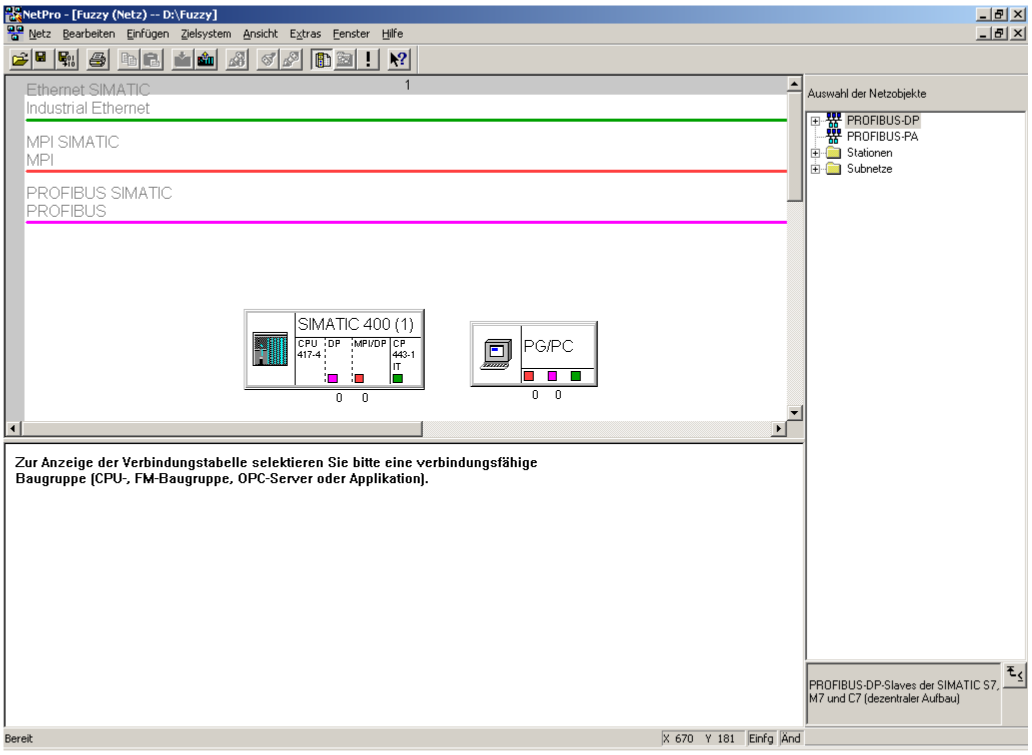
CP1613: 192.168.110.1
Subnet Mask: 255.255.255.0
Gateway: 192.168.110.1
Geben Sie bitte jeweils vier Zahlen - durch Punkte getrennt - im Bereich von 0-255 an.

- Now close the properties window and then the PG/PC-interface settings window by clicking on OK and confirm your settings in the following popup dialog box.

- You must now create an offline network configuration and load it into the SIMATIC. To do this, open the SIMATIC Manager and your project. In this example, the project is called D:\Fuzzy.



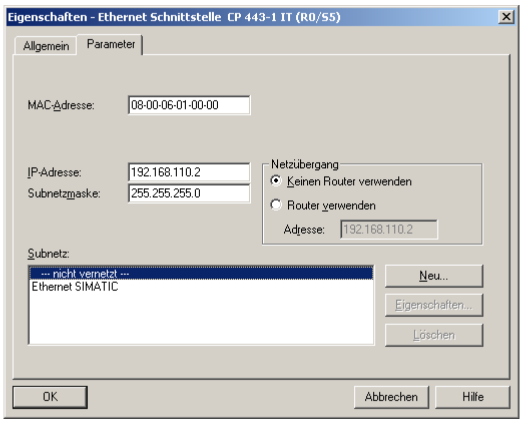
- Select a SIMATIC CPU and double-click on connections in the right-hand window. Upon this, the NetPro program opens with the current configuration.



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

- To reallocate a TCP/IP address to the SIMATIC, double-click on the Industrial Ethernet Interface (green box). Enter both target addresses e.g. 08-00-06-01-00-00 in MAC address and 192.168.110.2 in IP-address. In the lower Subnets window you can see, that in the current example that the interface is not yet networked. Therefore, click on the desired network, Ethernet SIMATIC in this case.



MAC-Adresse: 08-00-06-01-00-00

IP-Adresse: 192.168.110.2

Subnetzmaske: 255.255.255.0

Netzübergang

[✓ Keinen Router verwenden]

Router verwenden

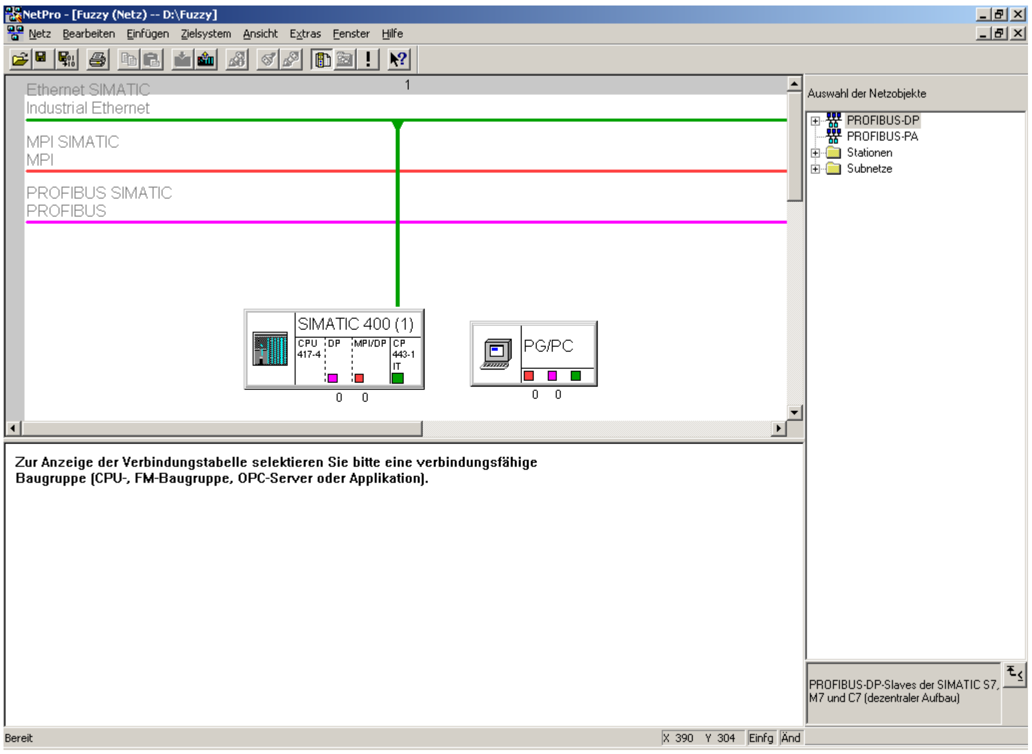
Adresse: 192.168.110.2

Subnetz:

--- nicht vernetzt ---

Ethernet SIMATIC

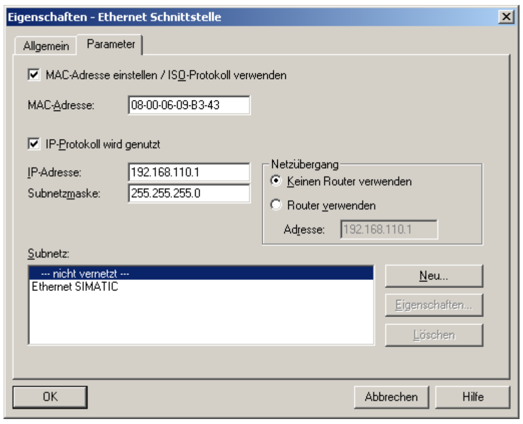
- Confirm your settings with OK. A connection between the SIMATIC and the Industrial Ethernet Bus is now displayed (green line).



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

- The PG/PC are put on the bus in a similar manner. Double-click the Industrial Ethernet interface (green box) and enter both addresses that were assigned in PG/PC settings. In this example they are 08-00-06-09-B3-43 in MAC-address and 192.168.110.1 in IP-address. Create the desired connection, by clicking on the corresponding Subnet, Ethernet SIMATIC in this case.


Eigenschaften - Ethernet Schnittstelle

Allgemein | Parameter

☑ MAC-Adresse einstellen / ISO-Protokoll verwenden

MAC-Adresse: 08-00-06-09-B3-43

☑ IP-Protokoll wird genutzt

IP-Adresse: 192.168.110.1

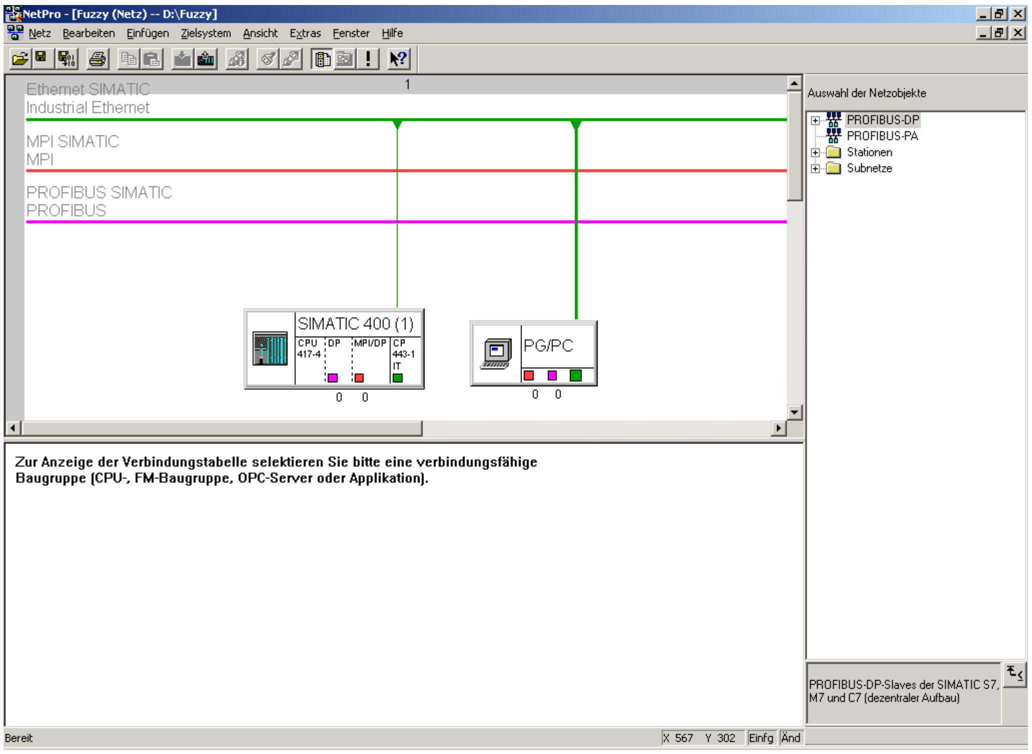
Subnetzmaske: 255.255.255.0

Netzübergang
- (●) Keinen Router verwenden
- ( ) Router verwenden

Adresse: 192.168.110.1

Subnetz:
- --- nicht vernetzt ---
- Ethernet SIMATIC

- After confirming your settings with OK, a connection between PG/PC and the Industrial Ethernet bus is displayed (green line).



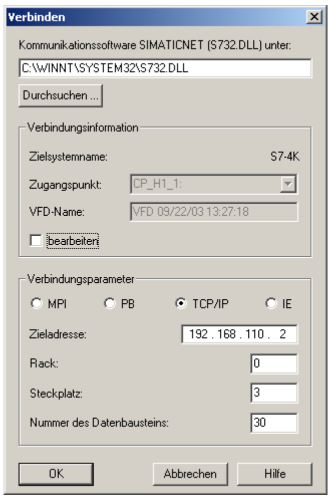
Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

- Lastly you must compile and load into the SIMATIC. To do this, click on ‘save and compile’ and then choose the upload option, after you have selected the desired SIMATIC.

- Now FuzzyControl++ can connect to the SIMATIC via TCP/IP. In the example, you must also select the CP_H1_1 access point and the TCP/IP radio button, enter the target address as 192.168.110.2 and click on OK in the FuzzyControl++ Connect dialog box.

- Rack, Slot and data block settings.
  CAUTION: The DB30 data block must be loaded into the S7 and initialized by the Fuzzy function blocks, i.e. it must be run at least once.


Verbinden

Kommunikationssoftware SIMATICNET (S732.DLL) unter: C:\WINNT\SYSTEM32\S732.DLL

Verbindungsinformation
Zielsystemname: S7-4K
Zugangspunkt: CP_H1_1:
VFD-Name: VFD 09/22/03 13:27:18
☐ bearbeiten

Verbindungsparameter
MPI PB [✓ TCP/IP] IE
Zieladresse: 192 . 168 . 110 . 2
Rack: 0
Steckplatz: 3
Nummer des Datenbausteins: 30

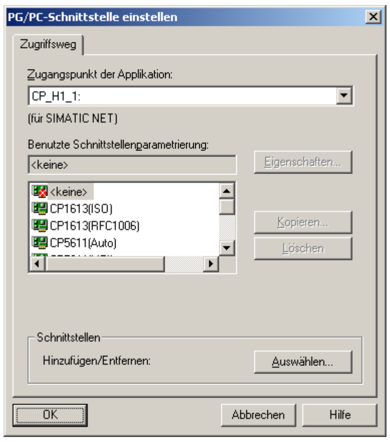
A.2.3.1 Overview of the configuration for an TCP/IP connection

PG/PC- interface	Access point: **CP_H1_1** -> CP1613 (RFC1006) PG/PC-MAC- address: 08-00-06-09-B3-43 PG/PC-IP- address: 192.168.110.1
NetPro	SIMATIC-MAC- address: 08-00-06-01-00-00 SIMATIC-IP- address: **192.168.110.2**

	PG/PC-MAC- address: 08-00-06-09-B3-43 PG/PC-IP- address: 192.168.110.1
FuzzyControl++	Access point: **CP_H1_1** Protocol: **TCP/IP** Target address: **192.168.110.2**

A.2.4 Creating an example configuration for an IE connection

- Firstly, open the PG/PC interface settings in the Control Panel.



PG/PC-Schnittstelle einstellen

Zugriffsweg

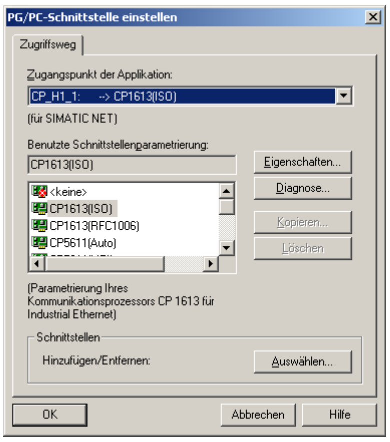
Zugangspunkt der Applikation: CP_H1_1:

(für SIMATIC NET)

Benutzte Schnittstellenparametrierung: <keine>

<keine>
CP1613(ISO)
CP1613(RFC1006)
CP5611(Auto)

- Most access points are not assigned to a component. Therefore, choose the desired access point from the upper list e.g. CP_H1_1 and the relevant component from the lower list e.g. CP1613 (ISO). After this, the assignment is shown in the form of an arrow in the upper field.


PG/PC-Schnittstelle einstellen

Zugriffsweg

Zugangspunkt der Applikation:

CP_H1_1: --> CP1613(ISO)

(für SIMATIC NET)

Benutzte Schnittstellenparametrierung:

CP1613(ISO)

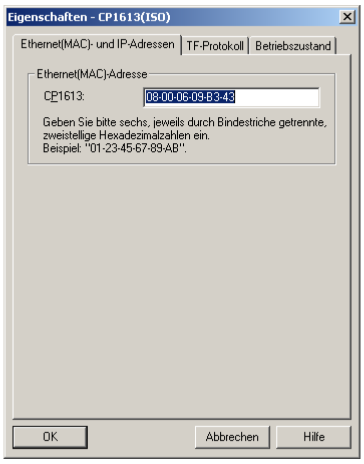
<keine>
CP1613(ISO)
CP1613(RFC1006)
CP5611(Auto)

(Parametrierung Ihres Kommunikationsprozessors CP 1613 für Industrial Ethernet)

Schnittstellen

Hinzufügen/Entfernen:

- The access point must now be appropriately configured. To do this select the access point to be parameterized now from the upper list, CP_H1_1 -> CP1613 (ISO) in this example, and click on the properties button. Enter the desired MAC address of the CP in CP1613 in the Ethernet (MAC) section, e.g. 08-00-06-09-B3-43.


Eigenschaften - CP1613(ISO)

Ethernet(MAC)- und IP-Adressen | TF-Protokoll | Betriebszustand

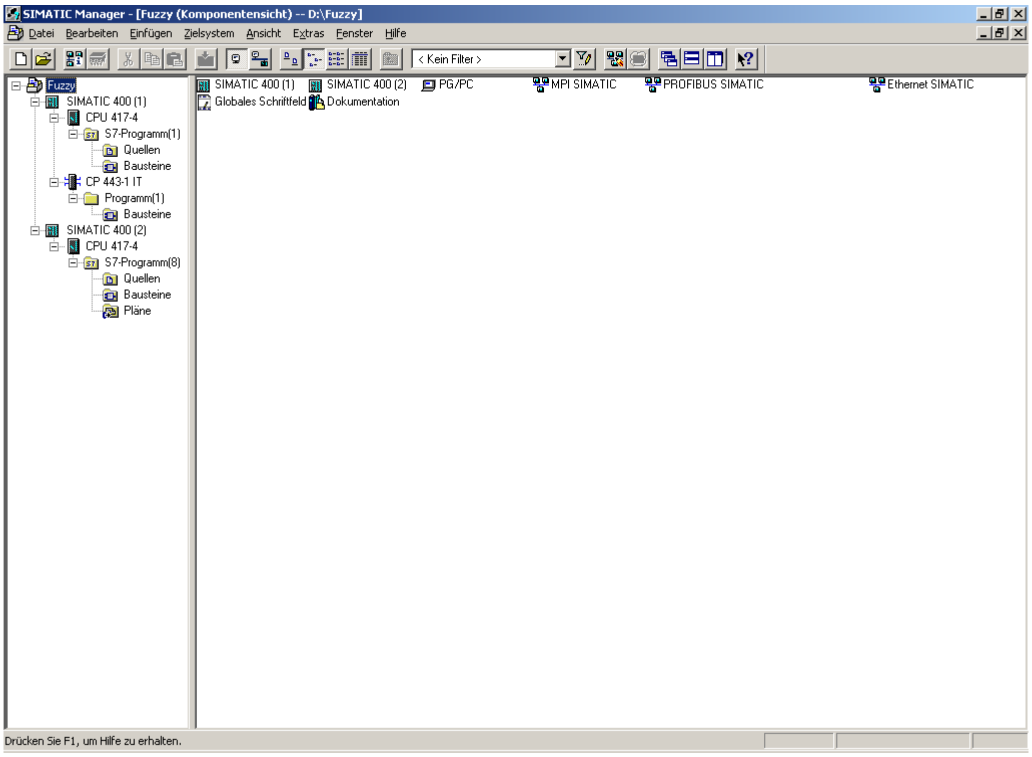
Ethernet(MAC)-Adresse

CP1613: 08-00-06-09-B3-43

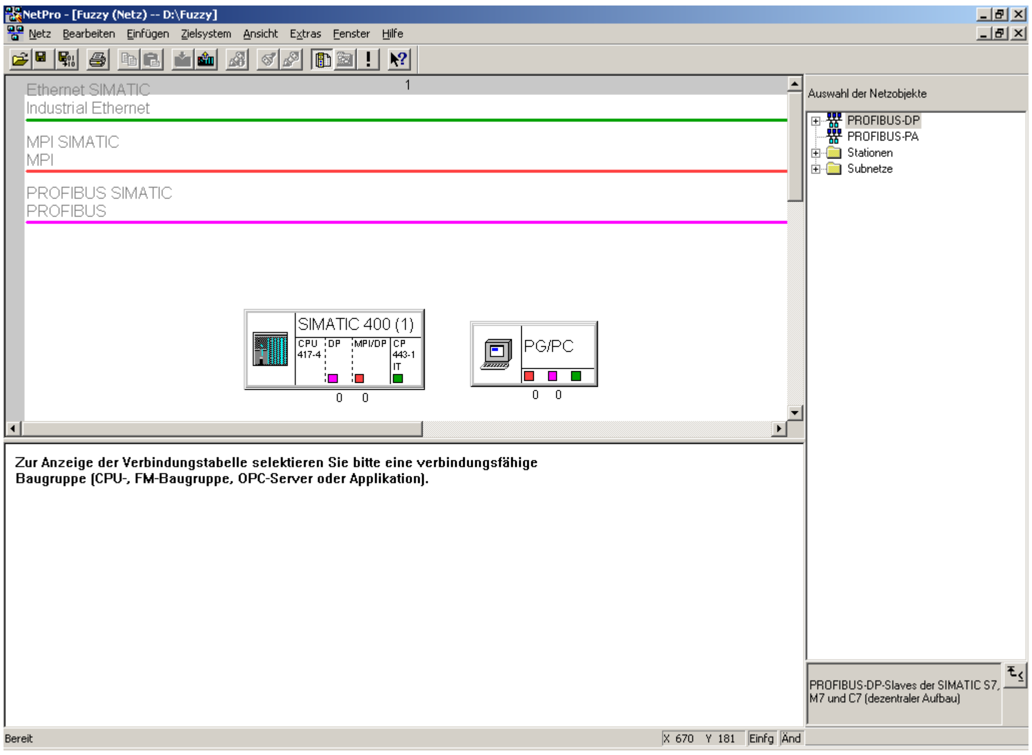
Geben Sie bitte sechs, jeweils durch Bindestriche getrennte, zweistellige Hexadezimalzahlen ein.
Beispiel: "01-23-45-67-89-AB".

- Now close the properties window and then the PG/PC-interface settings window by clicking on OK and confirm your settings in the following popup dialog box.

- You must now create an offline network configuration and load it into the SIMATIC. To do this, open the SIMATIC Manager and your project. In this example, the project is called D:\Fuzzy.

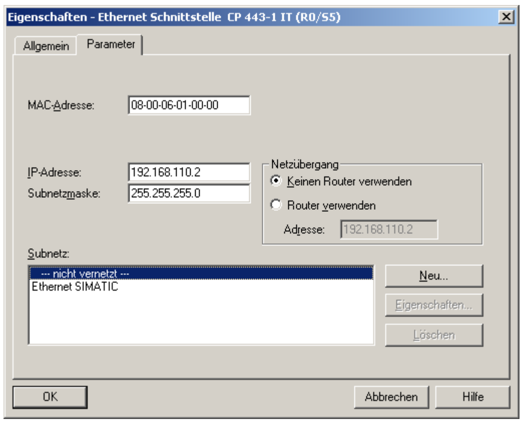


- Select a SIMATIC CPU and double-click on connections in the right-hand window. Upon this, the NetPro program opens with the current configuration.



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

- To reallocate an IE address to the SIMATIC, double-click on the Industrial Ethernet Interface (green box). Enter both target addresses e.g. 08-00-06-01-00-00 in MAC address and 192.168.110.2 in IP-address. In the lower Subnets window you can see, that in the current example that the interface is not yet networked. Therefore, click on the desired network, Ethernet SIMATIC in this case.



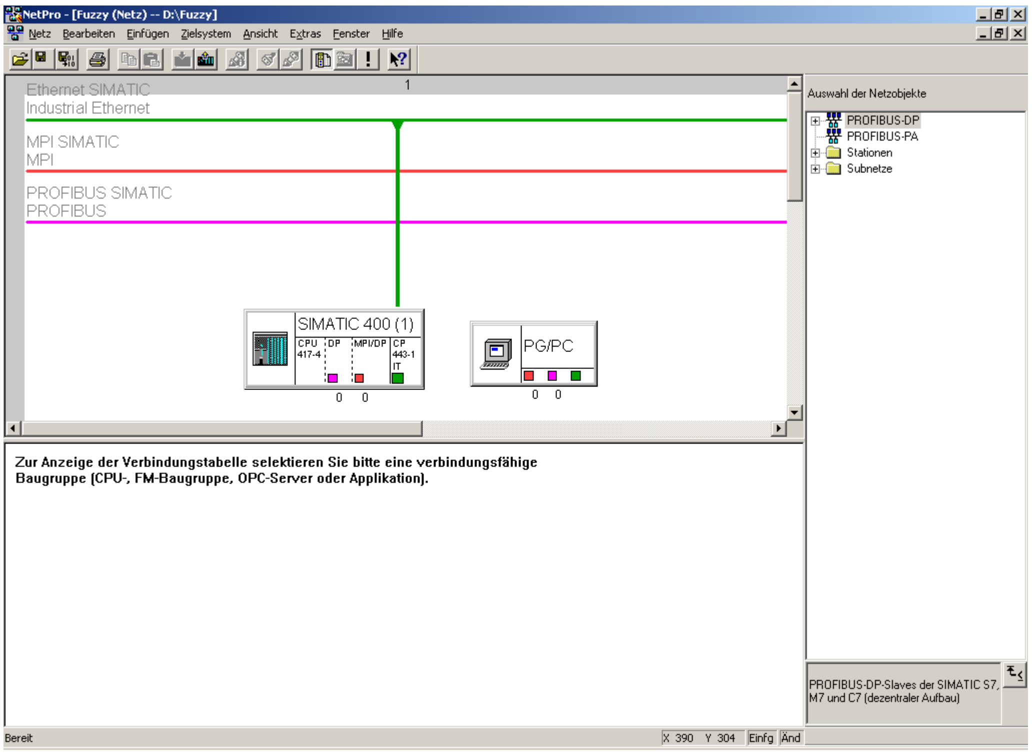
MAC-Adresse: 08-00-06-01-00-00

IP-Adresse: 192.168.110.2
Subnetzmaske: 255.255.255.0

Netzübergang
(●) Keinen Router verwenden
( ) Router verwenden
Adresse: 192.168.110.2

Subnetz:
--- nicht vernetzt ---
Ethernet SIMATIC

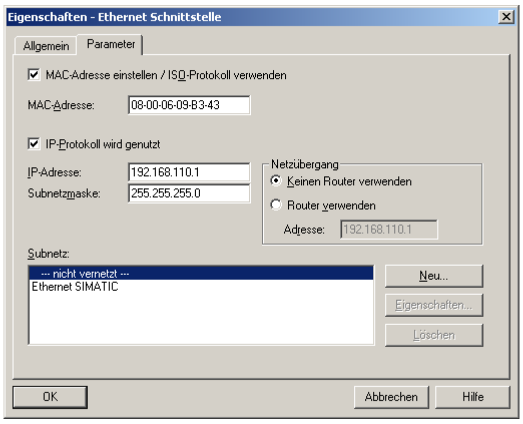
- Confirm your settings with OK. A connection between the SIMATIC and the Industrial Ethernet Bus is now displayed (green line).



Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

PROFIBUS-DP-Slaves der SIMATIC S7, M7 und C7 (dezentraler Aufbau)

- The PG/PC are put on the bus in a similar manner. Double-click the Industrial Ethernet interface (green box) and enter both addresses that were assigned in PG/PC settings. In this example they are 08-00-06-09-B3-43 in MAC-address and 192.168.110.1 in IP-address. Create the desired connection, by clicking on the corresponding Subnet, Ethernet SIMATIC in this case.


Eigenschaften - Ethernet Schnittstelle

Allgemein | Parameter

☑ MAC-Adresse einstellen / ISO-Protokoll verwenden

MAC-Adresse: 08-00-06-09-B3-43

☑ IP-Protokoll wird genutzt

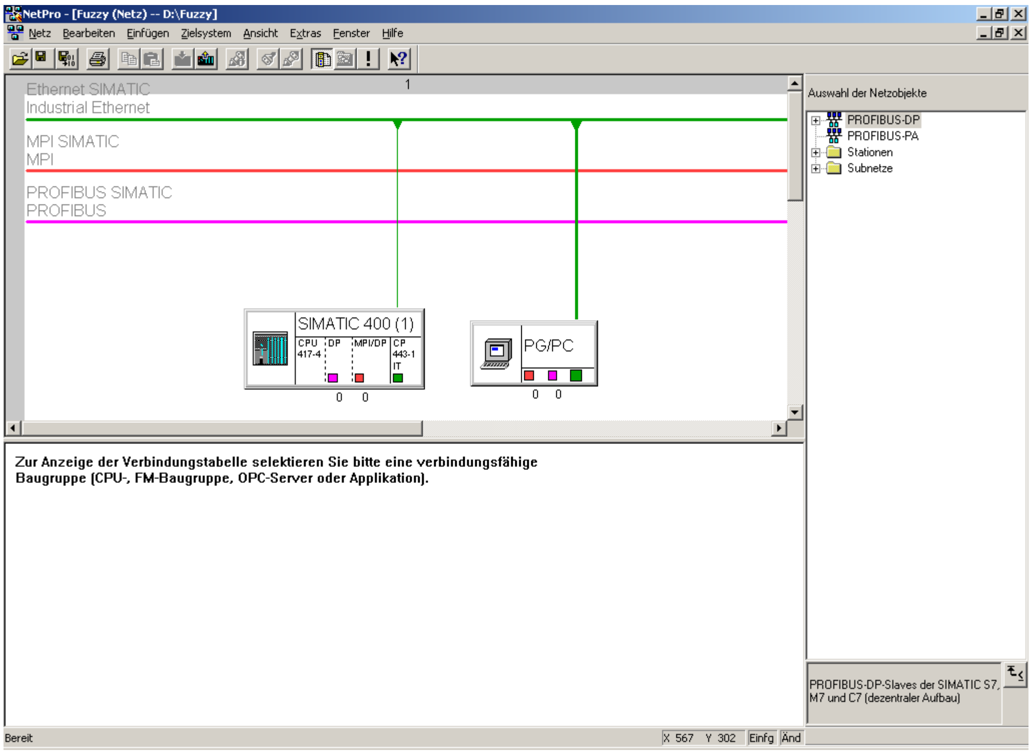
IP-Adresse: 192.168.110.1

Subnetzmaske: 255.255.255.0

Netzübergang
(●) Keinen Router verwenden
( ) Router verwenden
Adresse: 192.168.110.1

Subnetz:
--- nicht vernetzt ---
Ethernet SIMATIC

- After confirming your settings with OK, a connection between PG/PC and the Industrial Ethernet bus is displayed (green line).

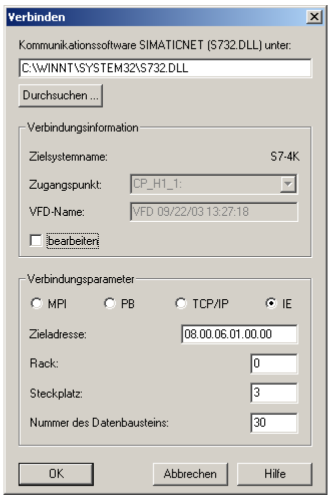


Zur Anzeige der Verbindungstabelle selektieren Sie bitte eine verbindungsfähige Baugruppe (CPU-, FM-Baugruppe, OPC-Server oder Applikation).

- Lastly you must compile and load into the SIMATIC. To do this, click on 'save and compile' and then choose the upload option, after you have selected the desired SIMATIC.

- Now FuzzyControl++ can connect to the SIMATIC via Industrial Ethernet. In the example, you must also select the CP_H1_1 access point and the IE radio button, enter the target address as 08.00.06.01.00.00 and click on OK in the FuzzyControl++ Connect dialog box.

- Rack, Slot and data block settings.
  CAUTION: The DB30 data block must be loaded into the S7 and initialized by the Fuzzy function blocks, i.e. it must be run at least once.


Verbinden

Kommunikationssoftware SIMATICNET (S732.DLL) unter:
C:\WINNT\SYSTEM32\S732.DLL

Verbindungsinformation
Zielsystemname: S7-4K
Zugangspunkt: CP_H1_1:
VFD-Name: VFD 09/22/03 13:27:18
☐ bearbeiten

Verbindungsparameter
MPI PB TCP/IP [✓ IE]
Zieladresse: 08.00.06.01.00.00
Rack: 0
Steckplatz: 3
Nummer des Datenbausteins: 30

A.2.4.1 Overview of the configuration for an IE connection

PG/PC- interface	Access point: **CP_H1_1** -> CP1613 (ISO) PG/PC-MAC- address: 08-00-06-09-B3-43 PG/PC-IP- address: 192.168.110.1
NetPro	SIMATIC-MAC- address: **08-00-06-01-00-00** SIMATIC-IP- address: 192.168.110.2

	PG/PC-MAC- address: 08-00-06-09-B3-43 PG/PC-IP- address: 192.168.110.1
FuzzyControl++	Access point: **CP_H1_1** Protocol: **IE** Target address: **08.00.06.01.00.00**

REFERENCES

[1] C. von Altrock: Fuzzy Logic, Volume 1: Technologie; R. Oldenbourg Verlag; Munich, Vienna; 1993

[2] M. Black: Vaguess: An Exercise in Logical Analysis; Philosophy of Science 50; pp.427-455; 1937

[3] R. Brause: Neuronale Netze: Eine Einführung in die Neuroinformatik; B.G. Teubner Verlag; Stuttgart; 1995

[4] A. Cichocki, R. Unbehauen: Neural Networks for Optimization and Signal Processing; B.G. Teubner Verlag; Stuttgart; 1993

[5] W. S. McCulloch, W. Pitts: A logical calculus of the ideas immanent in nervous activity; Bulletin of Mathematical Biophysics 5; pp.115-133; 1943

[6] D. Driankov, H. Hellendorn, M. Reinfrank: An Introduction to Fuzzy Control; Springer-Verlag; Berlin, Heidelberg; 1993

[7] K. A. Flaton: Neuronale Netze: Grundlagen und Anwendungen; Reihe: Neuro+Fuzzy; Zimmermann (Hrsg.); VDI-Verlag GmbH; Düsseldorf; 1995

[8] S. Hafner: Neuronale Netze in der Automatisierungstechnik; R. Oldenbourg Verlag, Munich, Vienna; 1994

[9] D. O. Hebb: The organization of behavior; John Wiley; New York; 1949

[10] N. Hoffmann: Kleines Handbuch Neuronale Netze; Vieweg & Sohn Verlag; Brunswick; 1993

[11] J. Kahlert: Fuzzy Control für Ingenieure: Analyse, Synthese und Optimierung von Fuzzy-Regelungssystemen; Vieweg & Sohn Verlag GmbH; Brunswick; 1995

[12] H. Kiendl: Fuzzy Control methodenorientiert; R. Oldenbourg Verlag; Munich, Vienna; 1997

[13] G. J. Klir, B. Yuan: Fuzzy Sets and Fuzzy Logic: Theory and applications; Prentice Hall Inc.; New Jersey; 1995

[14] M. Koch, Th. Kuhn, J. Wernstedt: Fuzzy Control: Optimale Nachbildung und Entwurf optimaler Entscheidungen; R. Oldenbourg Verlag; Munich, Vienna; 1996

[15] T. Kohonen: Self-organized formation of topologically correct feature maps; Biological Cybernetics; pp. 59-69; 1972

[16] B. Kosko: Neural Networks and Fuzzy Systems: A Dynamical Systems Approach to Machine Intelligence; Prentice Hall, Inc.; New Jersey; 1992

[17] J. Lukasiewicz, A. Tarski: Untersuchungen über den Aussagenkalkül; Comptes Rendus Soc. Sci. et Lettr. Varsovie III, 23; pp. 30-50; 1932

[18] D. E. Rummelhart, G. E. Hinton, R. J. Williams: Learning internal representations by error propagation; Rummelhart/McClelland (ed.), Parallel distributed, Processing: Explorations in the Microstructure of Cognition 1; MIT Press; pp. 318-362; 1986

[19] Siemens AG: Grundlagen der Fuzzy-Logik: Handbuch *FuzzyControl* für SIMATIC S7; pp. 1-2; 1995

[20] Siemens AG Automation Group: Fuzzy-Systeme in Theorie und Anwendungen Fuzzy aus den Erfahrungen der Siemens AG; Nuremberg; 1997

[21] M. Sugeno: Industrial Applications of Fuzzy Control; North-Holland-Amsterdam; London; 1985

[22] M. Sugeno: An Introductory Survey of Fuzzy Control; Information Sciences 36; pp. 59-83; 1985

[23] T. Terano, K. Asai, M. Sugeno: Applied Fuzzy Systems; AP Prof.; Boston; 1994

[24] VDI; VDE-Gesellschaft Meß- und Automatisierungstechnik: Neuronale Netze: Anwendungen in der Automatisierungstechnik; VDI Berichte 1184; VDI-Verlag GmbH; Düsseldorf; 1995

[25] P. J. Werbos: Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences; Diss.; Harvard University; Cambridge; 1974

[26] T. Wolf, Siemens AG: Optimization of Fuzzy Systems using Neural Networks and Genetic Algorithms; Proceedings of the 2nd European Congress of Intelligent Techniques and Soft Computing EUFIT; pp.544-551, Aachen; 1994

[27] L. Zadeh: Fuzzy Sets; Information and Control 8; pp.338-353; 1965

[28] A. Zell: Simulation neuronaler Netze; Addison-Wesley Publishing Company; Bonn; 1994

[29] H.-J. Zimmermann: Fuzzy Technologien: Prinzipien, Werkzeuge, Potentiale; VDI-Verlag GmbH; Düsseldorf; pp. 203-218; 1993

INDEX

NOTES