A DSP-BASED FUZZY SYSTEM FOR REAL-TIME INTERFERENCE CANCELLATION IN DS-SS SYSTEMS.

Joan Bas ,Xavier Alberti, Ana Pérez-Neira[1]
Department of Signal Theory and Communications.
Universitat Politècnica de Catalunya.
C/Jordi Girona 1-3, Mòdul D5
Campus Nord UPC 08034 Barcelona (Spain)
Tel: +34 93 4016454; fax: +34 93 4016447. e-mail : { jbas,anuska}@gps.tsc.upc.es

Abstract

This work demonstrates that high-end classical programmable digital signal processors (DSPs) can implement fuzzy systems for real-time communication processing. A systematic procedure to program the DSP is also described. A Fuzzy Narrowband Interference Canceller (NBI) has been designed and its performance compared when its membership functions are triangular or gaussian. Furthermore a comparative study with other cancellers has been carried out.

Keywords: *DSP, TMSC670, Interference Canceller, Fuzzy, Spread Spectrum.*

1. Introduction

Recently, non-linear signal processing has been found to be an attractive alternative in many fields of application. Regrettably, whenever a non-linear, possibly fuzzy, solution may be favoured, its actual implementation can be a problem. Many recent investigations and products (e.g. [1-2]) aim at coupling speed with complexity. Actually, a high-performance inference-dedicated circuit can be too expensive. In this paper we describe how a commercial DSP can be used to implement a fuzzy system for real-time purposes. Specifically, we focus on the problem of suppressing impulse noise in a Spread Spectrum wireless communication. Today, wireless communications is the largest single market segment for programmable DSPs. For the new generation wireless personal assistants flexibility is becoming more of an issue, and therefore the programmability offered by DSPs is even more desirable. We study the possibilities of a last-generation floating-point DSP by Texas Instrument which is inspired in the Very Long Instruction Word paradigm (VLIW). In general, the choice between fixed- and floating-point DSPs must take many factors into account, the most important being system precision and dynamic range, power consumption, and cost. Typically, fixed-point DSPs are used where low system cost and power consumption is paramount, whereas floating-point processing are preferred when the application requires higher precision and dynamic range. Floating-point DSPs can also provide a time-to-market advantage when data scaling and support for high-level languages are issues. Specifically, this work employs the TMSC6701 Texas Instruments DSP.

The paper structure is the following: firstly, Section 2 presents the Fuzzy system design, describing the interrelation among all parts of a fuzzy system. Next, Section 3 comments on the fuzzy canceller filter application and shows some simulation results. Finally conclusions come.

2. Fuzzy system design.

A Fuzzy system consists on four parts (see Fig. 1): Fuzzification, Inference, Rule Base and Defuzzification. If we want to implement a real-time application over a DSP we have to design all fuzzy parts jointly to get the best ratio between computational burden and time cost. In the following subsections we analyse the relationships between each of the fuzzy system components.

2.1. Fuzzification.

The Fuzzification is the interface between the crisp and fuzzy domain. In *Singleton* fuzzification, a key role is played by the input fuzzy sets. They give the degree in which a crisp data belong to the subsets that form a fuzzy variable. Physically the fuzzy sets are modelled by mathematical functions. These functions are normalised to the unity and are in charged to get the membership degrees. Many different types of membership functions can be used, although the more usual are the triangular and gaussian ones. The choice between triangular and Gaussian membership functions depends on the quality to time cost trade-off [3].

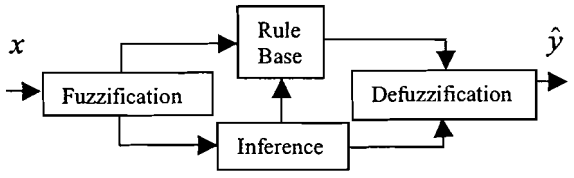

Fig. 1. Parts in a Fuzzy Logic System.

[1] This work has been supported by Spanish Government Grants: TIC 99-0849 and FIT-070000-2000-649 (Medea and Unilan European project) and The European Commission IST-1999-11729 METRA.

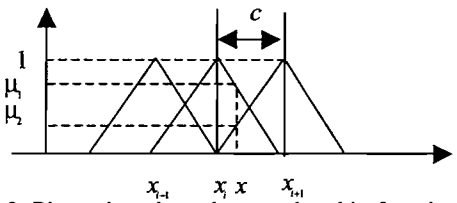

Fig. 2. Piecewise triangular membership functions.

Depending on the application, Gaussian membership functions offer better results than triangular ones. However, its computational burden sometimes it is too high to carry out a real-time implementation. Therefore, the use of linear functions can be a solution although it is possible to take advantage of the DSP architecture too. The interference canceller designed has been supported by the TMS320C6701 Texas Instruments 64-bits-floating point DSP, which has a clock of 167Mhz, a Very Large Instructions Word (VLIW) architecture, eight functional units (6 ALUs, 2 Multiplicative), 32 registers (A0-A15,B0-B15), 256 bit fetch packets containing 8 instructions that may be executed in parallel, one on each different unit, and pipeline techniques. Next, we list some points that are useful to reduce the computational burden that presents a fuzzy system application over a DSP.

1. Piecewise linear membership functions: this design implies that only two membership functions are activated at the same time. In addition, the sum of two membership degrees is the unity. (See Fig 2)

$$\mu_1(x) = \frac{x - x_i}{c} \quad (1)$$

$$\mu_2(x) = 1 - \mu_1(x) = 1 - \frac{x - x_i}{c} \quad (2)$$

where :

$\mu_1(x)$: membership degree of the first activated set

$\mu_2(x)$: membership degree of the second set

x_i : it is the mean of the i^{th} membership function.

c : it is the separation between two means.

Note that to calculate the membership degree (1) and (2) for each input fuzzy variable it is necessary to carry out one division and two subtractions. However, one division on TMSC6701 can spend 25 cycles, and therefore it is a serious drawback to implement it in a real-time application. Section 2.3 will comment on how it is possible to overcome this problem.

2. The DSP subroutine that is in charge of the fuzzification function is designed to get all membership degrees of one input fuzzy variable with a unique subroutine call. In this way we save time in its calling process.

3. Programming in assembler language to avoid/reduce the cycles that are spent in the software running.

4. Use of all TMSC670 DSP functional units. This allows, in the maximum case, carry out eight different operations at same time.

5. Use the pipeline technique to take advantage of the NOP cycles between two instructions. In this way, it is possible to start to process the following instruction before the current one had finished.

6. TMSC6701 DSP allows to condition all instructions to avoid the high delay of Branching (6 cycles, 5 delay + 1 evaluation).

7. Improve the memory access loading or storing two words simultaneously.

8. Use all DSP registers instead of memory addresses to save the data. This is based on that the register fetches spend less time to data access than memory fetches. In the TMS320C6701 Texas Instruments we have used their 32 registers (A0-A15,B0-B15) to allocate the program variables.

Finally, note that the number of membership functions that are activated simultaneously affects directly to the complexity of the inference block.

2.2. Rules.

The Rules relate the input membership functions with the output ones by means IF THEN statements as (3) show:

$$R_i \rightarrow If \;\; x_1 \; is \;\; A_1^i \; and \cdots x_m \; is \;\; A_m^i \; Then \;\; y \;\; is \;\; B^i \quad (3)$$

Where :

x_j ,y: they are the j^{th} input and output fuzzy variables.

R_i : it stands for the i^{th} fuzzy rule.

A_j^i, B^i : they are the j^{th} input and output fuzzy sets for the i^{th} rule

If in a Fuzzy system we have N input variables, and each one has P membership functions, then, the total number of possible rules is $M = P^N$. However, not always all rule combinations are possible. In the application commented in this paper, the fuzzification block only activates two input membership functions per input. Therefore, the real number of rules that finally are fired is 2^N. In the proposed DSP implementation, the rules have been coded as a matrix of M rows by (N+1) columns. The rows represent the number of rules and the columns represent all the inputs and outputs present in the fuzzy system. This matrix is important because it contains all inference parameters. Therefore, the faster we access to it, the faster we will calculate the inference block. One way to get a faster access is to look for the index of the input membership functions that are repeated. Once the repetitions have been found, we take one input fuzzy

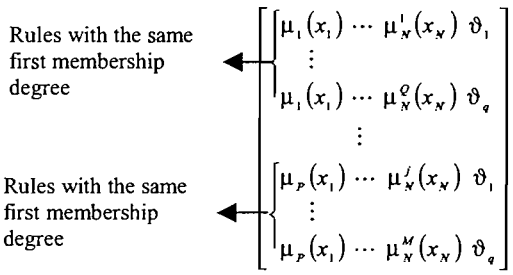

Fig.3. Rule base matrix order.

Where:

$\mu_k(x_i)$: it represents the input membership function that has been taken as a reference.

$\mu_i^j(x_i)$: it represents the membership function degree of the i^{th} input (i: 2.N) and the j^{th} fuzzy rule (j:1.M)

ϑ_j : it represents the output centroid of the j^{th} fuzzy rule (j:1.M).

p : number of blocs with the same input membership function degree.

variable as a reference. Next we sort the inference matrix according to the same membership function index repeated (see Fig. 3). Furthermore, in the fuzzification step if we use piece-wise linear functions there are only two membership functions activated at the same time. In this way it is possible to increase the access speed to the inference matrix. At programming level, each group of rules have been coded in assembler and identified by a label (memory address). Nevertheless, not all the groups or blocks have to be processed to carry out an inference because only two membership degrees per input fuzzy variable are activated simultaneously. Therefore designing some strategies for the access to the inference matrix would be useful to get a real-time application. In this paper we propose two strategies. The first one considers that only two membership functions are activated per fuzzy input (Optimisation I). The second takes advantages of the inference matrix sorting (Optimisation II). Finally, the obtained inference values are processed in the defuzzificaton step.

2.3. Defuzzification.

The defuzzification stage converts a fuzzy variable into a crisp variable. In this work we have used the centroid method which calculates the centre of gravity (COG) of all rules that have been inferred. Its formulation is the following:

$$\hat{y} = \frac{\sum_{j=1}^{M} \theta_j \prod_{i=1}^{n} \mu(x_i)_j}{\sum_{j=1}^{M} \prod_{i=1}^{n} \mu(x_i)_j} \qquad (4)$$

In the centroid defuzzifier we can rewrite, (4) as follows:

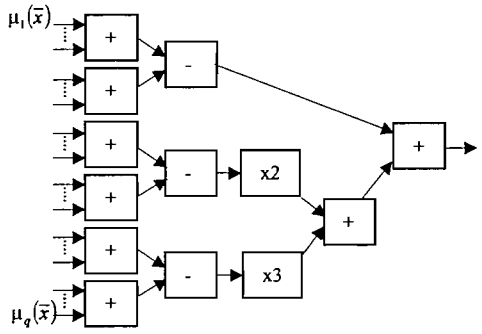

Fig. 4. Architecture of an inference block.

$$\hat{y} = \frac{\sum_{j=1}^{M} \theta_j \prod_{i=1}^{n} \frac{\beta(x_i)_j}{c}}{\sum_{j=1}^{M} \prod_{i=1}^{n} \frac{\beta(x_i)_j}{c}} = \frac{\sum_{j=1}^{M} \theta_j \prod_{i=1}^{n} \beta(x_i)_j}{\sum_{j=1}^{M} \prod_{i=1}^{n} \beta(x_i)_j} \qquad (5)$$

Where $\beta(x)$ can be: $\beta(x) = x - x_i$ or $\beta(x) = c - (x - x_i)$. Therefore, choosing the centroid defuzzifier technique we only have to carry out two subtractions per membership degree. Next, we group the rules that have the same output centroid. Then, (5) can be rewritten as (6):

$$\hat{y} = \frac{\theta_1 \sum_{j=1}^{P} \prod_{i=1}^{n} \beta(x_i)_j + \cdots + \theta_r \sum_{j=M-L+1}^{M} \prod_{i=1}^{n} \beta(x_i)_j}{\sum_{j=1}^{P} \prod_{i=1}^{n} \beta(x_i)_j + \cdots + \sum_{j=M-L+1}^{M} \prod_{i=1}^{n} \beta(x_i)_j} \qquad (6)$$

In this way it is possible to reduce the number of products that there are in the numerator of (5). In addition, note that the sums in the numerator and denominator of (6) are the same. Therefore, it is not necessary to recalculate twice these sums. This fact has been exploited in its implementation using parallel instructions and pipeline techniques.

3. Application to interference cancellation.

The system that has been implemented cancels a narrow band interference that corrupts a spread spectrum signal. For a detailed description we refer to [4]. In the designed system we have 3 inputs, 7 membership functions by input fuzzy variable and 72 effective rules. The Fuzzification is *Singleton*, the inference is Sugeno and its operators are product and sum, the defuzzification method is the Centre of Gravity (COG) where the output centroids are {-3,-2 ...3}. Applying (6) and grouping the centroids that have the same absolute value, the defuzzification function results in:

$$\hat{y} = \frac{\left(\sum_{j=1}^{P} \prod_{i=1}^{n} \beta(x_i)_j - \sum_{i=Q}^{r} \prod_{i=1}^{n} \beta(x_i)_i\right) + \cdots + 3\left(\sum_{j=1}^{P} \prod_{i=1}^{n} \beta(x_i)_j - \sum_{i=Q}^{r} \prod_{i=1}^{n} \beta(x_i)_i\right)}{\sum_{j=1}^{P} \prod_{i=1}^{n} \beta(x_i)_j + \cdots + \sum_{j=M-L+1}^{M} \prod_{i=1}^{n} \beta(x_i)_j} \qquad (7)$$

The rule base matrix that implements (7), has been grouped in 7 blocks, because the input membership

functions have 7 membership functions. However, only two blocks will be fired simultaneously, because only two input fuzzy sets will be activated. The analysis of these two blocks will contribute to calculate the numerator and denominator of (7) cumulatively. The architecture of one of these blocks is depicted in Fig. 4. To show the importance of the membership functions two fuzzy systems have been designed. The first uses triangular membership functions (Fuzzy-T), whereas the second uses Gaussian ones (Fuzzy-G). The Fuzzy systems performance has been compared with the linear and non-linear interference cancellers of [6-7]. Two sided linear filters which weights are adapted by a LMS algorithm (denoted by TS-LMS) compose the linear method [6]. The non-linear method was proposed by [7] and it is a modification of an adaptive Approximate Conditional Mean (ACM) filtering algorithm, being its acronym DRD2. Table I and II show the efficiency of interference cancellers measuring the SNR improvement when the interference signal is an autoregressive process with two poles in z=0.99 (Table I), or when it is a 1 tone with digital frequency w=0.15 (Table II). The number of chips simulated was 2500, the processing gain was 10 and 10 Montecarlo runs were averaged.

If the interference to cancel is autoregressive, the fuzzy system compared with the TS-LMS and DRD2 systems offers similar or better efficiency. However, when the interference is 1 tone the fuzzy system performances are between that those of TS-LMS and DRD2. Nevertheless, the fuzzy systems are not adapted whereas the other two systems need to be adapted to estimate the interference parameters; thus increasing the complexity. Furthermore, from Table I and Table II we can conclude that the SNR improvement when the membership functions are gaussian or triangular is very similar, although there is an important computational load reduction if we use triangular membership functions.

Fig. 5, shows the computational burden of the fuzzy system designed with triangular membership functions (Fuzzy-T) when we apply the different implementation strategies commented in Section 2 (No optimisation, Optimisations I and II). Next, Table III shows some temporal specifications about Fuzzy-T system, like the time spent to process a sample or its Mega Inferences per Second (MIPS).

4. Conclusions

The proposed canceller architecture allows selecting a reduced number of rules that form the fuzzy system. In this way, the computational burden of a complex system can be highly simplified, being affordable and attractive for commercial applications. To reduce the computational load a rule minimisation criteria, such as that proposed in [5] has to be further studied.

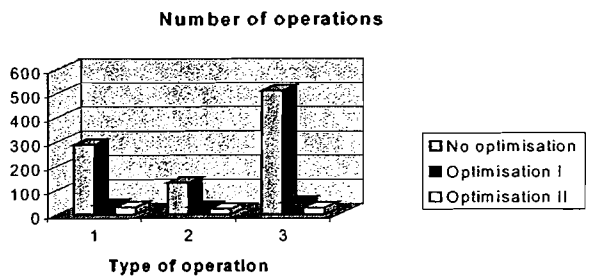

Fig. 5 Number of operations for each type of optimisation.(1-Mult.,2.- Sums/subs,3.- Memory access).

Table I. SNR improvement for AR interference.

Method\SIR(dB)	-20	-15	-10	-5
Fuzzy-T	36	31.9	27.1	21.6
Fuzzy-G	36.1	31.9	26.8	21.9
TS-LMS	26.9	22.3	17.6	13
DR2D	37	32.7	28.2	23.4

Table II. SNR improvement for 1 tone interference.

Method\SIR(dB)	-20	-15	-10	-5
Fuzzy-T	22.7	26.8	23.3	20.2
Fuzzy-G	24.3	26.8	24.1	20.2
TS-LMS	27.7	23.2	18.1	13
DR2D	38.5	33.6	28.7	23.8

Table III. Fuzzy-T system specifications.

Operation	Specifications
Fuzzification	60 cycles
Inference+ Defuzzification	115 cycles
Total cycles number	175 cycles
Total time	0.702μs/sample
Mega Inferences per Second (MIPS)	20.95 MIPS

References

1. R.Rovatti,M.Vittuari,"Linear and fuzzy piecewise-linear signal processing with an extended DSP architecture", IEEE World congress on Computational Intelligence, pp.1082-1087,Alaska,May 1998
2. R. Rovatti, A. Ferrari, M. Borgatti, "Automatic Implementation of Piecewise-Linear Fuzzy Systems Addressing Memory-Performance Trade-Off", FUZZY HARDWARE: Architectures and Applications, Edit: A. Kandel, G. Langholz Kluwer Acad. Pub., Boston, 1998
3. J.Bas, A.Pérez,"Fuzzy Adaptive Signal Predistorter for OFDM Systems", X European Signal Processing Conference pp. Finland,September 2000
4. A.Pérez-Neira, J. Cid,Sueiro, J. Roca, M.A. Lagunas,"A dynamic non-singleton fuzzy logic system for DS/CDMA communications," FUZZ-IEEE'98, Proceedings pp. 1494-1499, May 1998, Anchorage
5. R.Rovatti,R.Guerreri,T.Villa,"Fuzzy Rules Optimization for Analog VLSI Implementation",ATIP95 : IEEE Fuzzy, March 1995 Yokohama.
6. C.J.Masreliez,"Approximate non-Gaussian filtering with linear state and observation relations",IEEE Transactions on Automatic Control, February 1975
7. W.Wu,F. Yu,"New nonlinear algorithms for estimating and suppressing narrow-band interference in DS-SS systems," IEEE Transactions on Communications, vol. 44,April 1996