VHDL-AMS Based Genetic Optimization of Mixed-Physical-Domain Systems in Automotive Applications

Leran Wang
Tom J. Kazmierski
School of Electronics and Computer Science
University of Southampton
Southampton, SO17 1BJ
United Kingdom
lw04r@ecs.soton.ac.uk

This paper presents a VHDL-AMS based genetic optimization methodology suitable for performance improvement of hardware systems in automotive applications. Models of such systems are mixed-signal (analog and digital) in which the analog parts cover mixed physical domains. A case study applying this novel method to the fuzzy logic controller (FLC) optimization in an automotive active suspension system (AASS) has been investigated. A new type of fuzzy logic membership functions with variable geometrical shapes has been proposed and optimized. In this optimization technique, VHDL-AMS is used not only for the modeling and simulation of the FLC and its underlying AASS but also for the implementation of a parallel genetic algorithm (GA). This has resulted in an integrated performance optimization system wholly implemented in the hardware description language (HDL). Results show that the proposed FLC has superior performance to that of existing FLCs that use fixed-shape membership functions.

Keywords: VHDL-AMS, fuzzy logic controller, genetic algorithm, active suspension

1. Introduction

This paper presents a general approach to complex hardware system optimization using a hardware description language. Traditionally, hardware systems are optimized using dedicated software applications which repeatedly invoke a suitable hardware description language (HDL) simulator [1]. The interaction between the external program and the simulator normally requires multiple data transfers (Figure 1(a)). The salient feature of the technique presented here is that the hardware description testbench includes a genetic algorithm (GA) optimizer which concurrently simulates multiple instances of the system (chromosomes). In this way, both the hardware system and the optimizer are integrated within a single software platform (Figure 1(b)). The GA optimizer presented here is implemented in VHDL-AMS. It was successfully applied to a case study where the performance of a fuzzy logic controller (FLC) in an automotive active suspension system (AASS) is optimized by varying the shapes of its membership functions. The VHDL-AMS model of the AASS is complex and it involves mechanical, hydraulic and electrical blocks which interact with each other.

Genetic algorithms are statistical optimization methods which use techniques inspired by natural selection [2, 3]. Some basic GA elements include: chromosome, gene, fitness and population. Chromosomes present candidate solutions to certain problems, i.e. the hardware system to be optimized in this case. Genes are the variables within a chromosome, such as the design parameters for a hardware system. Fitness is associated to each chromosome to determine its performance. A number of chromosomes form the population, which will evolve over time through a process of competition and controlled variation [4]. A GA process usually consists of selection, crossover

SIMULATION, Vol. 85, Issue 10, October 2009 661–670

DOI: 10.1177/0037549709106693
Figure 8 appears in color online: http://sim.sagepub.com

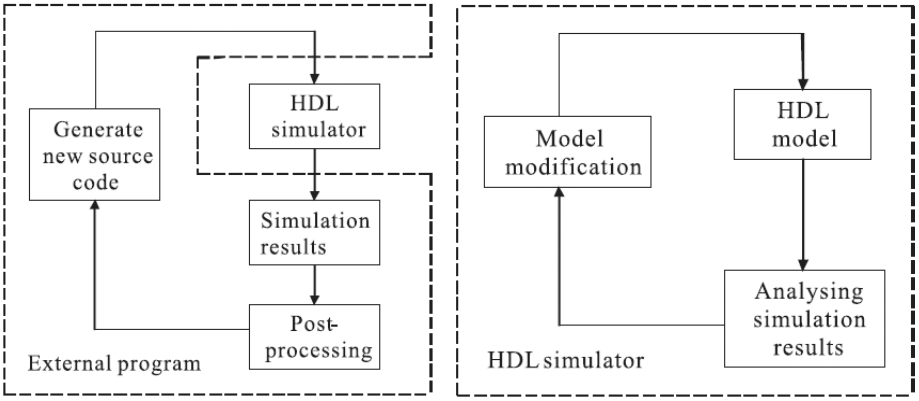

Figure 1. Comparison of optimization approaches. (a) Traditional hardware systems optimization. (b) Proposed integrated modeling and optimization

and mutation. Selection is the stage in which a proportion of the existing population is selected to breed a new generation. Individual chromosomes are selected through a fitness-based algorithm, where fitter chromosomes are typically more likely to be selected. Crossover is a genetic operator used to vary the genes of a chromosome to produce the next generation. Usually a pair of 'parent' chromosomes will exchange part of their genes to form a pair of 'child' chromosomes. Mutation is a genetic operator that introduces new genes into the population. This is often done by altering a small number of existing genes with random values [5]. This selection-crossover-mutation process is repeated until all the chromosomes converged to an optimal or a certain optimization goal has been achieved.

Automotive suspension systems reduce vibrations between the sprung and unsprung masses caused by the motion of an automobile so that the vehicle's ride quality could be improved. According to the suspension system's ability to add or extract energy, they can be classified as passive, semi-active or active [6]. Passive suspension consists of conventional springs and dampers and it cannot add energy to the system. Semi-active suspension does not add energy either but changes the damping coefficients of the shock absorbers dynamically to obtain a better suspension quality. In contrast, active suspension contains an actuator, which adds energy to the system by generating a force on the sprung and unsprung masses, as well as the springs and dampers.

The advantages of active suspension systems over passive and semi-active ones have been known for many years [7]. However, the design of a suitable AASS controller, which determines the value of the actuator force according to the dynamic motions of the sprung and/or unsprung mass, is difficult and keeps attracting researchers' interest. A number of different control algorithms have been established [6, 8–10]. However, an accurate performance analysis and optimization of such systems is still difficult since the input to an AASS (i.e. the road displacement) is unpredictable. Fuzzy logic controllers, due to their ability of handling uncertain and complex systems, have emerged as a promising technique for high-performance control of AASS [11–13]. FLCs are based on general principles of the fuzzy set theory [14] and their input and output variables are similar to a conventional controller. Each input variable will first be converted to a linguistic variable with certain degrees of truth according to fuzzy membership functions. These linguistic variables then pass through the fuzzy inference procedure to generate the output linguistic variable. The output linguistic variable will be converted back to FLC's output by the defuzzification process [12]. FLC designs reported so far show satisfactory suspension behavior and use fixed-shape, usually triangular or trapezoidal membership functions. We report here how variable-shape membership functions, coupled with effective optimization, can significantly improve the overall AASS performance.

VHDL-AMS [15] has been chosen to implement both the model and underlying optimizer, because this hardware description language was specifically designed to support modeling at various abstraction levels in mixed, electrical and non-electrical physical domains as well as mixed, digital and analog components [16]. Moreover, it has been recommended by an automotive industry consortium as a unified modeling language for automotive applications [17]. Also, the concurrent nature of VHDL-AMS makes implementation of GA optimization efficient and straightforward.

A GA has been reported to optimize various features of a fuzzy controller. For example, a GA was used to optimize the decision-making rules for fuzzy PI/PD controllers [18]. The input variables to an FLC can also be chosen by a GA [19]. A GA has also been used to tune the vertices of triangular membership functions of an FLC [20]. In the research presented in this paper, a GA is used to optimize not only the vertices but also the geometrical

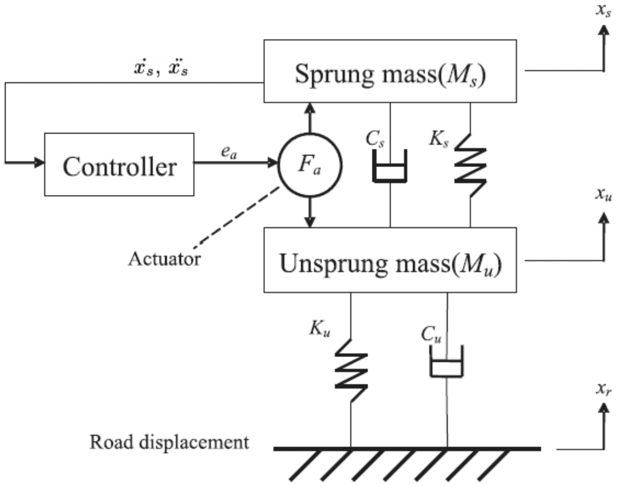

Figure 2. Active suspension system

shapes of the fuzzy logic membership functions to further improve an FLC's performance. The hardware description implementation presented here uses multiple instances of the automotive suspension system as populations of chromosomes implemented at the VHDL-AMS testbench level. The testbench also implements other GA elements, such as a selection process which rejects unfit chromosomes, a crossover to produce new offspring, and a random mutation of new offspring.

2. System Model

2.1 Suspension Model

Figure 2 shows a linear 2-DOF (degree of freedom) quarter-car model [21]. The dynamic behavior of the sprung and unsprung masses is described by equations (1) and (2) [10] which can be obtained from Newton's second law:

$$\ddot{x}_s M_s = K_s(x_u - x_s) + C_s(\dot{x}_u - \dot{x}_s) + F_a \tag{1}$$

$$\begin{aligned} \ddot{x}_u M_u = {} & -K_s(x_u - x_s) - C_s(\dot{x}_u - \dot{x}_s) \\ & + K_u(x_r - x_u) + C_u(\dot{x}_r - \dot{x}_u) - F_a \end{aligned} \tag{2}$$

where M_s and M_u are the vehicle's sprung and unsprung masses; x_s, x_u and x_r are the displacement of sprung mass, unsprung mass and road respectively; K_s and C_s are the coefficients of the passive spring and damper; K_u and C_u are the tyre stiffness and damping rate; and F_a is the actuator force.

2.2 Actuator Model

The actuator is an electro-mechanical hydraulic design [22], which operates in parallel with the passive springs and dampers. It consists of a DC motor, some mechanical parts (gear train and rack) and a hydraulic vibration absorber (Figure 3). The actuator input from the controller is a DC voltage (e_a), which drives the motor to output a rotational torque (T_m). The gear train converts the rotational torque into the translational velocity (V_r) of the rack, whose displacement impacts the pressures of the upper and lower chambers of the hydraulic piston (P_1 and P_2, respectively). The unsprung mass is attached to the hydraulic cylinder's piston rod and the sprung mass sits on the cylinder cap. The actuator force (F_a) is dependent on the pressure difference between the upper and lower chambers. The generated force attenuates the vibrations by acting on the sprung and unsprung masses.

The motor develops a torque proportional to the armature current (i_a):

$$T_m = K_t i_a. \tag{3}$$

When the armature rotates, the back electro-magnetic flux (EMF) induces a voltage proportional to the angular velocity:

$$e_b = K_b \frac{d\theta_{g1}}{dt}. \tag{4}$$

The circuit can be analyzed using Kirchoff's voltage law:

$$L_a \frac{di_a}{dt} + R_a i_a + e_b = e_a. \tag{5}$$

Newton's second law is applied to the input dynamics (J_{l1}) as follows:

$$J_{l1} \frac{d^2\theta_{g1}}{dt^2} + b_{l1} \frac{d\theta_{g1}}{dt} + T_{g1} = T_m. \tag{6}$$

The gear train is considered ideal so that the relationship between the input and output angular velocity (ω_{g1} and ω_{g2}) and torque (T_{g1} and T_{g2}) can be written as:

$$\frac{\omega_{g2}}{\omega_{g1}} = \frac{r_{g1}}{r_{g2}} = \frac{n_{g1}}{n_{g2}} \quad \text{and} \quad T_{g1}\omega_{g1} = T_{g2}\omega_{g2}. \tag{7}$$

Application of Newton's second law to the load shaft (J_{l2}) yields:

$$J_{l2} \frac{d^2\theta_{g2}}{dt^2} + b_{l2} \frac{d\theta_{g2}}{dt} + T_L = T_{g2}. \tag{8}$$

The rack linear velocity (V_{r2}) can be determined from:

$$V_{r2} = \omega_{g2} r_{g2} \quad \text{and} \quad V_{r2} = V_{r1}. \tag{9}$$

The load torque (T_L) acting on the load shaft is:

$$\begin{aligned} & T_L = F_r r_{g2} \\ \text{where} \quad & F_r = A_{r2}P_2 - A_{r1}P_1 + M_r a_r - M_r g. \end{aligned} \tag{10}$$

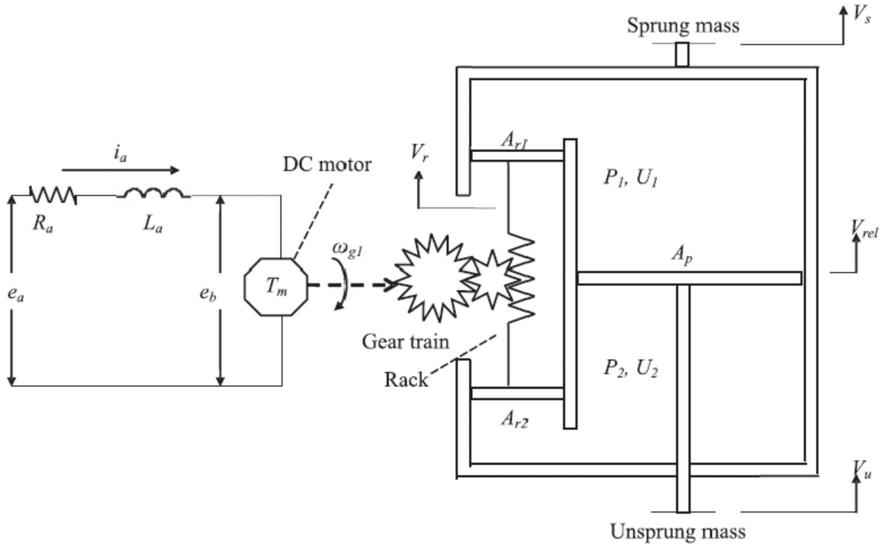

Figure 3. Actuator model for the AASS

Table 1. Fuzzy rules base

		Acceleration		
		P	**Z**	**N**
Velocity	**P**	N	N	Z
	Z	N	Z	P
	N	Z	P	P

The hydraulic pressure in the upper and lower actuator chambers are described as:

$$\frac{dP_1}{dt} = (+A_p V_{rel} - A_{r1} V_{r1}) \left(\frac{\beta_1}{U_1}\right) \tag{11}$$

$$\frac{dP_2}{dt} = (-A_p V_{rel} + A_{r2} V_{r2}) \left(\frac{\beta_2}{U_2}\right). \tag{12}$$

The actuator force (F_a) is given by:

$$F_a = (P_1 - P_2) * A_p. \tag{13}$$

The above differential algebraic equations (DAEs) describe the behavior of the actuator components and these equations cover electrical, mechanical and hydraulic domains.

2.3 FLC Model

It has been reported that the optimum choice for active suspension's state variables are the velocity and acceleration of the automobile sprung mass M_s [23]. So these two variables are chosen as the inputs to the FLC. The output is the DC voltage driving the actuator e_a. The fuzzy sets of the input and output variables are represented by three linguistic variables: *positive* (P), *zero* (Z) and *negative* (N), for which a set of nine fuzzy rules is developed, as shown in Table 1. The fuzzy inference procedure used is the max-product composition [24]. Assuming that the sprung mass velocity has the degree of membership v_P, v_Z and v_N in *positive* (P), *zero* (Z) and *negative* (N) respectively and the sprung mass acceleration has the degree of membership a_P, a_Z and a_N, the *positive* degree of the output force F_a is:

$$F_P = \max(a_N{*}v_N, a_N{*}v_Z, a_Z{*}v_N). \tag{14}$$

Similarly, the *zero* and *negative* degree of F_a are:

$$F_Z = \max(a_N{*}v_P, a_Z{*}v_Z, a_P{*}v_N) \tag{15}$$

$$F_N = \max(a_P{*}v_P, a_Z{*}v_P, a_P{*}v_Z). \tag{16}$$

The defuzzification method employed is the centre of gravity approach [12]. The output force is calculated as:

$$F_a = F_{a\max}{*}(F_P - F_N)/(F_P + F_Z + F_N). \tag{17}$$

3. Shape Optimization of Fuzzy Logic Membership Functions

In fuzzy logic theory, a membership function is a graphical representation of the input's degree of participation in a fuzzy set. The geometrical shapes of the membership functions used can seriously affect the performance of an FLC. For example, although triangular membership functions are very basic and widely used in active suspension controllers [11, 13], it was also illustrated that

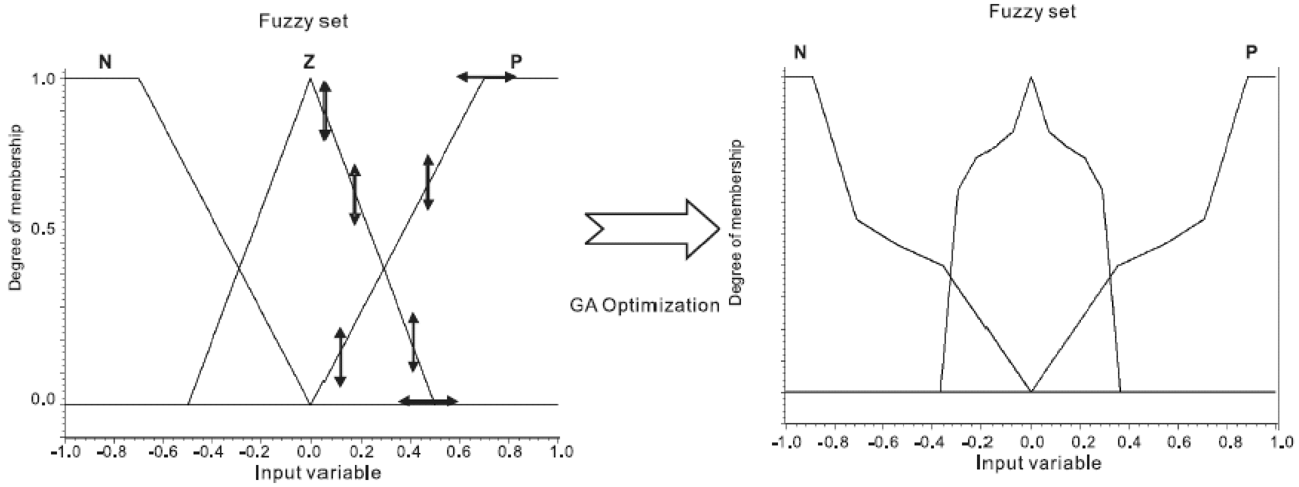

Figure 4. Optimization of the shapes of fuzzy logic membership functions

trapezoidal membership functions may generate superior results in certain applications [12]. The optimal shapes of membership functions in an FLC for a specific application may be irregular and we show below that they can be calculated by optimization to improve the system's performance (Figure 4). Irregular membership functions are unlikely to lead to more complex hardware implementations given the fact that electronic control units are quite common in today's automobile design.

In the GA optimization, instances of the AASS, including the sprung/unsprung mass, the spring and damper, the mixed-physical-domain actuator, and the FLC, are invoked and each instance represents a chromosome (Figure 5). The shapes of the fuzzy logic membership functions of the FLC are the optimization objects. The shapes of the fuzzy logic membership functions are determined by the positions of a number of points on the curve. Thus these points will represent the genes of each chromosome (Figure 4). Since the centre of gravity method is used for defuzzification, it is only necessary to optimize the membership functions of the input variables. As can be seen in Section 2.3, Equation 17, the centre of gravity defuzzification does not involve membership function of the output variable. For each of the two input variables, N points from the *positive* curve and N points from the right half of the *zero* curve are selected as genes (Figure 4). This is because a membership function is typically symmetrical about the y-axis. These points are equally distributed along the x-axis and their y-values can be adjusted between 0 and 1. These points are simply connected by straight lines to form piecewise linear membership functions. The optimization goal is to minimize the peak-to-peak value of the sprung mass displacement x_s when the system is subject to some kind of stimulus, i.e. x_{pp} is the chromosome's fitness value.

4. Parallel GA in VHDL-AMS Testbench

In the VHDL-AMS implementation, the chromosome is modeled as a component with *4N* genes as input parameters, the road displacement x_r as the excitation and the peak-to-peak value x_{pp} as the output fitness. Since x_{pp} is a value over a certain time period, a process is needed to track its maximum and minimum value and output the peak-to-peak value at the end. Figure 5 is the block diagram of the chromosome. It shows how different components in the VHDL-AMS entity are connected.

In the actuator part, the quantities in the upper row are 'through' quantities and the ones in the lower row are 'across' quantities. The VHDL-AMS code of the actuator is given below. It models the behavior of the actuator described in Section 2.2.

```
library ieee_1076_1_1;
use ieee_1076_1_1.mechanical_systems.all;
use ieee_1076_1_1.electrical_systems.all;
use ieee_1076_1_1.fluidic_systems.all;
use work.VibIso_System.all;
entity Actuator is
   port(quantity ea: in VOLTAGE; --applied
armature voltage(V)
   quantity Vrel: in VELOCITY; --relative
velocity of piston(m/s)
   quantity Fa: out FORCE --output force(N));
end entity Actuator;
architecture Behavior of Actuator is
   quantity Tm: TORQUE; --motor torque(N*m)
   quantity ia: CURRENT; --armature current(amp)
   quantity eb: VOLTAGE; --back emf(V)
   quantity omegag1,omegag2: ANGULAR VELOCITY;
     --angular velocity of gear(rad/s)
   quantity Tg1,Tg2: TORQUE; --torque applied
     on gear(N*m)
```

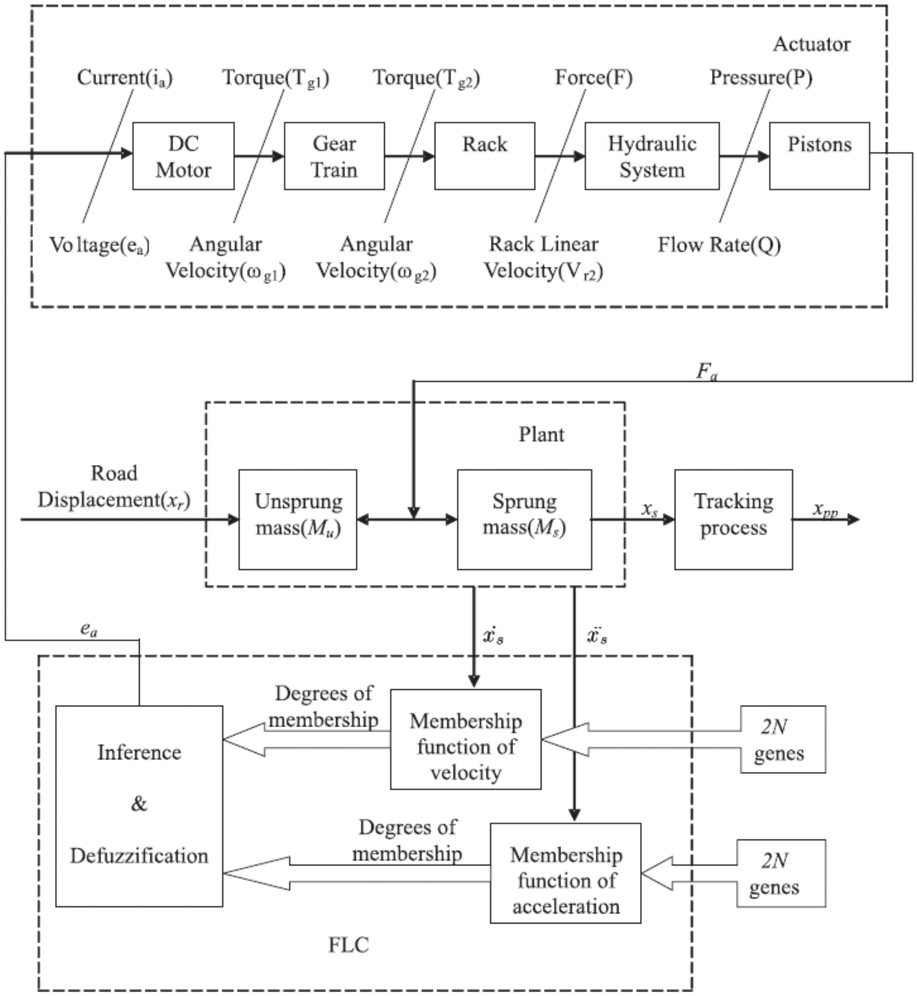

Figure 5. Diagram of one chromosome

```
   quantity TL: TORQUE; --load torque back to
     shaft(N*m)
   quantity Vr: VELOCITY; --linear velocity of
     rack(m/s)
   quantity P1,P2: PRESSURE; --chamber
     pressure(N/m^2)
   quantity U1,U2: VOLUME; --volume of
     chamber(m^3)
begin
   Tm == Kt*ia;
   eb == Kb*omegag1;
   ea == eb + Ra*ia + La*ia'DOT;
   Jl1*omegag1'DOT + bl1*omegag1 + Tg1 == Tm;
   omegag2 == N*omegag1;
   Tg1 == Tg2*N;
   Jl2*omegag2'DOT + bl2*omegag2 + TL == Tg2;
   TL == (Ar2*P2 - Ar1*P1)*rg2;
   Vr == omegag2*rg2;
   P1'DOT == (Ap*Vrel - Ar1*Vr)*(beta1/U1);
   P2'DOT == (-Ap*Vrel + Ar2*Vr)*(beta2/U2);
   U1'DOT == Ap*Vrel - Ar1*Vr;
   U2'DOT == -Ap*Vrel + Ar2*Vr;
   Fa == (P1 - P2)*Ap;
end architecture Behavior;
```

The actuator covers three different physical domains: electrical, mechanical and hydraulic. The VHDL-AMS code presented here incorporates components from these domains into one single model. Using the IEEE 1076.1.1 multiple energy domain standard packages [25], each quantity is defined by its physical name, such as torque, voltage, pressure, etc. These names are connected with their corresponding physical natures as shown in the code above.

A flow chart of how the parallel GA is implemented and executed in the VHDL-AMS testbench is shown in

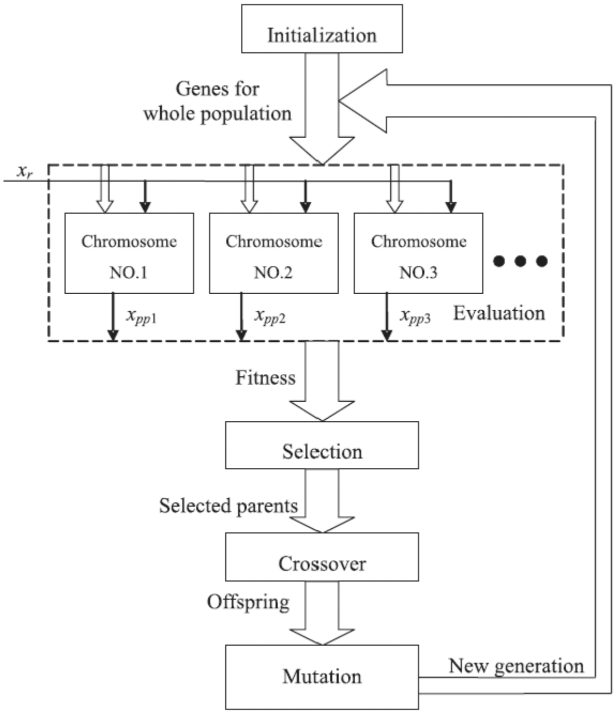

Figure 6. GA optimization in a VHDL-AMS testbench using concurrently running chromosomes

Figure 6. Unlike most existing computer implementations of GA that evaluate one chromosome iteratively to form a population, in the VHDL-AMS based optimization here, the chromosomes of a population are implemented in parallel. The genes are initialized by uniformly distributed random numbers. The same stimulus is applied to the population and all the chromosomes are evaluated simultaneously to get a vector of fitness values. According to the fitness values, the tournament selection is chosen as the parent's selection method because it prevents premature convergence with efficient computations [5]. The parent with higher fitness (i.e. smaller x_{pp}) is more likely to be selected to produce offspring. Elitism is also used to improve GA's performance by artificially inserting the best solution into each new generation. Since the genes are real numbers, arithmetic crossover is used to generate the offspring [26]. At last, gene mutation is employed to introduce new solutions into the new population. The evaluation-selection-crossover-mutation process is repeated until all the chromosomes converge to the same fitness. In VHDL-AMS, this loop is controlled by a finite state machine.

5. Experimental Results

In the GA optimization, the number of points on each membership curve N is chosen as 5. So there are totally 20 genes in one chromosome. The population size is 100. The crossover and mutation rate are 0.8 and 0.01 respectively. The stimulus is a single sine-wave period jolt with added Gaussian noise (GN). The sine-wave jolt is of a 10 cm amplitude and the period of 200 ms (5 Hz). The GN is formed from a wide-spectrum white Gaussian stochastic process with a 1 cm standard deviation passed through a 50 Hz low-pass filter. The formation of the stimulus is based on two considerations. First, for ride and handling characteristics the most important frequency range is 0.5–50 Hz, of which 5 Hz is the logarithmic middle value. Anything below 0.5 Hz is too small to cause any suspension deflection, while frequencies above 50 Hz are outside the bandwidth of tyre and suspension dynamics [27]. Second, the actual road displacement inputs are of a random nature, thus some pseudo-random noises have been added. The stimulus is repeated every four seconds, which is the system's settling time. The peak-to-peak value of x_s, x_{pp}, is also updated every four seconds as the chromosome's fitness. Simulations were carried out using the SystemVision [28] VHDL-AMS simulator from Mentor Graphics. After simulating the testbench for 800 seconds, which corresponds to 200 generations in the GA optimization, the shapes of the membership functions converge to an optimum (Figure 7). Figure 8 gives the fitness value of

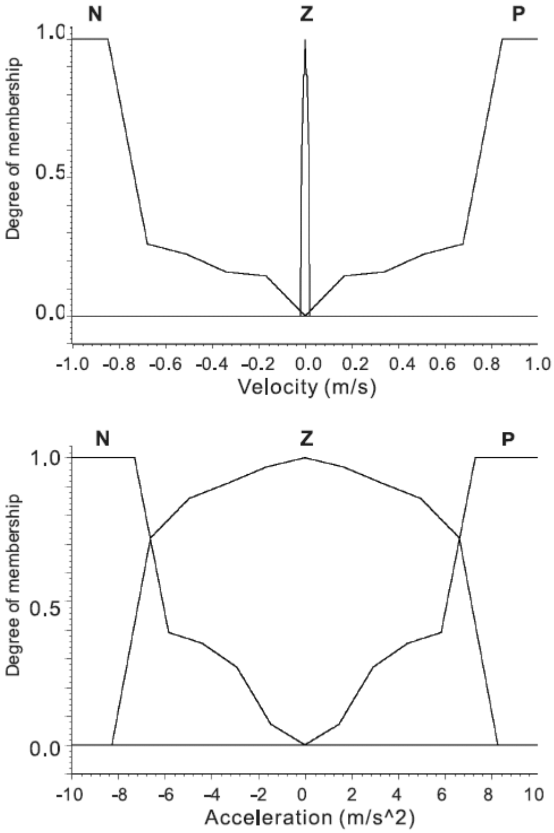

Figure 7. GA optimized membership functions

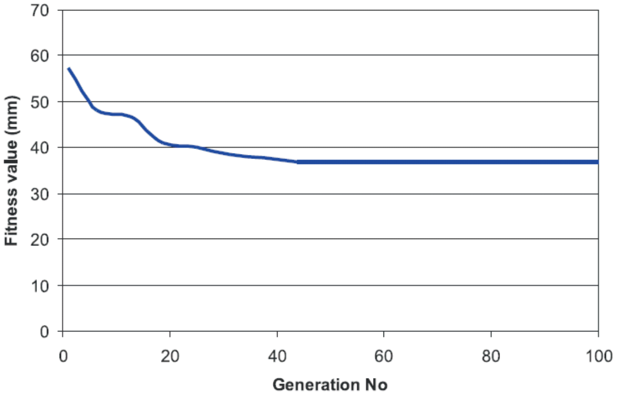

Figure 8. Convergence process of xpp

Table 2. Peak-to-peak and RMS values of responses to noisy jolt excitation (Case 1)

FLC type	xpp (mm)	RMS of xs (mm)
Trapezoidal	54.114	9.1851
Triangular	50.445	8.5145
GA optimised	36.240	5.9729

the best chromosome in the first 100 generations, which shows how the chromosomes are converged during the optimization. Because elitism is employed, the GA process converges quickly. The simulation CPU time was 31 hours 35 minutes and 50 seconds on a Pentium 4 PC and the runtime for each evaluation is about 5.7 seconds.

The GA-optimized membership functions are implemented in the FLC and simulated. For comparison, the FLCs using triangular and trapezoidal membership functions are also investigated. Three types of road displacement x_r have been used as system inputs. Simulation results of the three test cases are shown below:

Case 1: Noisy sine-wave jolt. In this test case, x_r is the same as the stimulus used in the GA optimization, a single 5 Hz sine-wave jolt with the GN. The simulation waveforms of three types of FLCs are shown in Figure 9. Table 2 lists the peak-to-peak values of x_s (x_{pp}) and the root mean square (RMS) values of x_s. The conventional FLCs can reduce x_{pp} from larger than 30 cm to 5.04 cm (triangular) and 5.41 cm (trapezoidal). The GA-optimized FLC developed here can further decrease the value to 3.62 cm, which is an improvement of more than 25%.

In the following two test cases, the GA optimized FLC is subjected to different types of stimulus to test the generalization performance of the GA optimization.

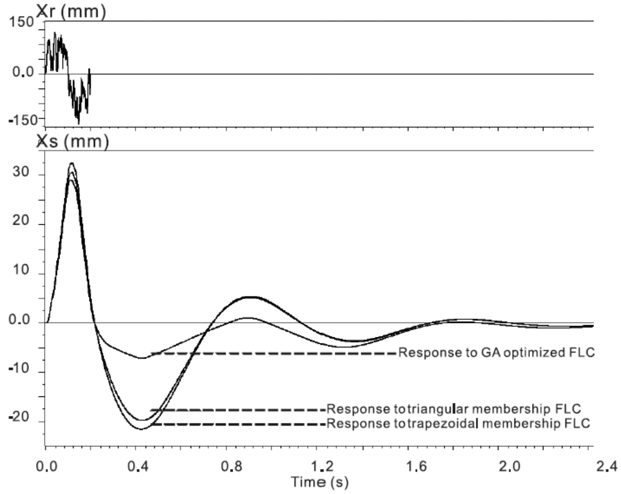

Figure 9. Waveforms for noisy jolt excitation (Case 1)

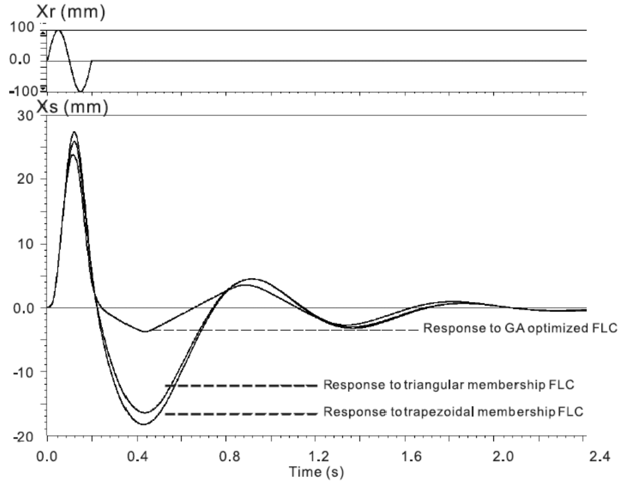

Figure 10. Waveforms for sine-wave jolt excitation (Case 2)

Table 3. Peak-to-peak and RMS values of responses to sine-wave jolt excitation (Case 2)

FLC type	xpp (mm)	RMS of xs (mm)
Trapezoidal	45.635	7.7675
Triangular	42.323	7.1325
GA optimised	27.563	4.6117

Case 2: Sine-wave jolt. The second x_r is a single 5 Hz sine-wave jolt of a 10 cm amplitude, which is of the same frequency as the stimulus used for optimization but without added noise. Simulation results are shown in Figure 10 and Table 3.

Case 3: Trapezoidal bump. The x_r is of a different shape from the stimulus used for optimization. The

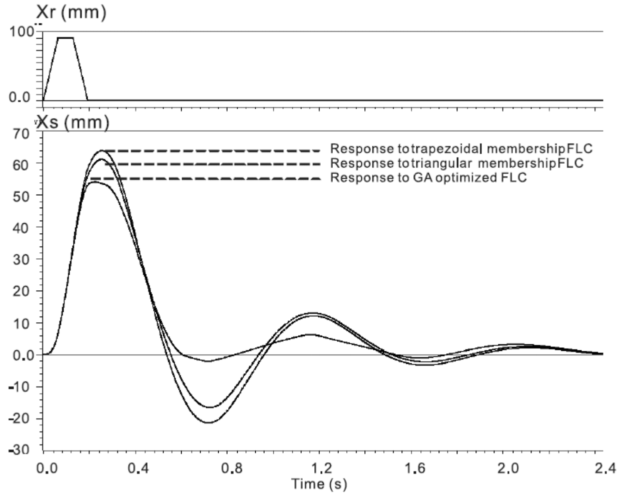

Figure 11. Waveforms for trapezoidal bump excitation (Case 3)

Table 4. Peak-to-peak and RMS values of responses to trapezoidal bump excitation (Case 3)

FLC type	xpp (mm)	RMS of xs (mm)
Trapezoidal	85.265	20.486
Triangular	77.695	19.503
GA optimised	56.223	17.116

trapezoidal bump is 10 cm high and 200 ms long in time. Simulation results are shown in Figure 11 and Table 4.

In all the above test cases, the GA-optimized FLC shows improvements in both the peak-to-peak and RMS values of sprung mass displacement to that of FLCs using trapezoidal and triangular membership functions. The results demonstrate that the proposed optimization method has good generalization performance.

6. Conclusion

This paper proposes a novel, integrated approach to complex hardware system optimization in which the optimizer is a part of the HDL-based simulation testbench. A VHDL-AMS implementation of a parallel GA was successfully used to optimize the shapes of fuzzy logic membership functions to improve the FLC's performance. A case study of the FLC for automotive active suspension systems has been investigated and the simulation results show that the GA-optimized FLC shows superior performance to that of conventional FLCs with triangular or trapezoidal membership functions.

7. References

[1] Hounsell, B.I. and T. Arslan. 2000. A Novel Genetic Algorithm for the Automated Design of Performance Driven Digital Circuits. In *Proceedings of the 2000 Congress on Evolutionary Computation*,1, 601–608.

[2] Holland, J.H. 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[3] Goldberg, D.E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Reading, MA.

[4] Herrera, F., M. Lozano, and J.L. Verdegay. 1998. Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis. *Artificial Intelligence Review* 12(4), 265–319.

[5] Mitchell, M. 1996. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA.

[6] Sam, Y.M., M.R.H.A. Ghani, and N. Ahmad. 2000. LQR Controller for Active Car Suspension. In *Proceedings of TENCON 2000,* 441–444.

[7] Tan, H.-S. and T. Bradshaw. 1997. Model Identification of an Automotive Hydraulic Active Suspension System. In *Proceedings of the American Control Conference,* 2920–2924.

[8] Alleyne, A. and J.K. Hedrick. 1995. Nonlinear Adaptive Control of Active Suspensions. *IEEE Transactions on Control Systems Technology* 3(1), 94–101.

[9] Yagiz, N., and I. Yuksek. 1997. Sliding Modes Control of Active Suspensions. In *Proceedings of the 1997 IEEE International Symposium on Intelligent Control*, 349–353.

[10] Chantranuwathana, S. and H. Peng. 2004. Adaptive Robust Force Control for Vehicle Active Suspensions. *International Journal of Adaptive Control and Signal Processing* 18(2), 83–102.

[11] Son, S.-I. and C. Isik. 1996. Fuzzy Control of an Automotive Active Suspension. In *Biennial Conference of the North American Fuzzy Information Processing Society*, 377–381.

[12] Barr, A.J. and J.I. Ray. 1996. Control of an Active Suspension using Fuzzy Logic. In *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*, 42–48.

[13] Al-Holou N., J. Weaver, T. Lahdhiri and D.S. Joo. 1999. Sliding Mode-based Fuzzy Logic Controller for a Vehicle Suspension System. In *Proceedings of the American Control Conference*, 4188–4192.

[14] Zadeh, L.A. 1965. Fuzzy Sets. *Information and Control*, 8, 338–353.

[15] *IEEE standard VHDL analog and mixed-signal extensions*. IEEE Std 1076.1-1999, 23 December 1999.

[16] Christen, E.and K. Bakalar. 1999. VHDL-AMS: A Hardware Description Language for Analog and Mixed-signal Applications. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 46, 1263–1272.

[17] Moser, E. and N. Mittwollen. 1998. VHDL-AMS: The Missing Link in System Design. Experiments with Unified Modelling in Automotive Engineering. In *Design, Automation and Test in Europe,* 59–63.

[18] Kuo, Y.-P. and T.-S.S. Li. 1999. GA-based Fuzzy PI/PD Controller for Automotive Active Suspension System. *IEEE Transactions on Industrial Electronics* 46(6), 1051–1056.

[19] Hashiyama, T., S. Behrendt, T. Furuhashi and Y. Uchikawa. 1995. Fuzzy Controllers for Semi-active Suspension System Generated through Genetic Algorithms. In *IEEE International Conference on Systems, Man and Cybernetics, 1995. 'Intelligent Systems for the 21st Century',* 5, 4361–4366.

[20] Moon, S.Y. and W.H. Kwon. 1996. Genetic-based Fuzzy Control for Automotive Active Suspensions. In *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*, 2, 923–929.

[21] Ulsoy, A.G., D. Hrovat, and T. Tseng. 1994. Stability Robustness of LQ and LQG Active Suspension. *Journal of Dynamic Systems, Measurement, and Control* 116, 123–131.

[22] Liu, X. and J. Wagner. 2002. Design of a Vibration Isolation Actuator for Automotive Seating Systems. *International Journal of Vehicle Design* 29(4), 335–375.

[23] Ro, P.I., C. Kim, and H. Kim. 1993. Active Suspension using Fuzzy Logic Control. In *Proceedings of the American Control Conference,* 2252–2253.

[24] Sugeno, M. 1985. An Introductory Survey of Fuzzy Control. *Information Sciences* 36, 59–83.

[25] Wilson, P.R., A.D. Brown, and H.A. Mantooth. 2003. Standard VHDL 1076.1.1 Packages for Multiple Energy Domain Support. In *Proceedings of the 2003 International Workshop on Behavioral Modeling and Simulation*, 70–75.

[26] Herrera, F., M. Lozano, and A.M. Sanchez. 2003. A Taxonomy for the Crossover Operator for Real-Coded Genetic Algorithms: An Experimental Study. *International Journal of Intelligent Systems* 18, 309–338.

[27] Truscott, A.J. and A.W. Burton. 1994. On the Analysis, Modelling and Control of an Advanced Automotive Suspension System. In *International Conference on Control'94*, 1, 183–189.

[28] Mentor Graphics Corporation. *SystemVision User's Manual*. Version 3.2, Release 2004.3. July 2004.

__Leran Wang__ received his M.Sc. degree from the University of Liverpool and Ph.D. degree from the University of Southampton, in 2004 and 2009 respectively. Before coming to the UK, he received the B.Eng. degree from Beijing University of Posts and Telecommunications in 2003. He is currently a visiting research fellow at the School of Electronics and Computer Science, University of Southampton, UK. His research interests focus on VHDL-AMS based modeling and optimization of mixed physical-domain systems for automotive, energy harvesting and MEMS applications.

__Tom Kazmierski__ (M'95) received his M.S. degree in electronic engineering from Warsaw University of Technology, Warsaw, Poland, in 1973 and his Ph.D. degree from the Military Academy of Technology, Warsaw, in 1976. Currently he is a Senior Lecturer in the School of Electronics and Computer Science, University of Southampton, UK, where he pursues research in numerical modeling, simulation, and synthesis techniques for computer-aided design of very large scale integration (VLSI) circuits. From 1990 to 1991, he was a Visiting Research Scientist with the IBM VLSI Technology Division, San Jose, CA, where he developed and patented synchronization techniques for multi-solver simulation backplanes. He has contributed to the development of the VHDL-AMS standard by IEEE, served as Chair of the IEEE DASC P1076.1 (VHDL-AMS) Working Group from 1999 to 2005. He has published over 100 papers and given a number of invited talks and tutorials mostly in the area of analog and mixed signal synthesis and HDLs. In recent years, he has been working on web-based electronic design frameworks and applications of VHDL-AMS to high-level system modeling and synthesis, involving modeling of mixed-domain systems, automated analog, and mixed-signal synthesis for ASIC design, including synthesis of artificial VLSI neural networks and performance modeling of mixed-technology electromechanical systems.