HARDWARE IMPLEMENTATION OF PLD-BASED FUZZY LOGIC CONTROLLERS USING A LOOK-UP TABLE TECHNIQUE

HENG SIEW TAN AND RICHARD SANDIGE
Department of Electrical Engineering
University of Wyoming

BOGDAN WILAMOWSKI
Department of Electrical Engineering
University of Wyoming

ABSTRACT:
This paper investigates a hardware implementation of a multi-input single-output fuzzy logic controller using a look-up table technique. This approach uses a programmable read only memory (PROM), that is, a programmable logic device (PLD), to realize the implementation. This concept is simple yet has a very fast response time for applications that require only a few inputs. This approach yields higher speeds compared to other fuzzy controllers that use a microprocessor. Simulation results for different resolutions are provided for the classical example of the backing truck to ramp problem. When using a look-up table the fuzzy logic's interpolation capabilities are shown to work well with a fairly low number of bits.

INTRODUCTION

A fuzzy controller can be designed either by programming a microprocessor system or by a digital hardware system composed of application specific integrated circuits (ASICs) or PLDs. Many current designs use a microprocessor system. Such software implementations of fuzzy logic controllers are accomplished using a microcontroller, a personal computer, or a workstation. Software implementations usually require several milliseconds to several hundreds of milliseconds to obtain the inference results. Processing time is further increased when the size of the knowledge base or the number of input variables is increased due to the large amount of real-time data processing required. Since fuzzy controllers are real-time systems, speed is important; the higher the speed the better the system.

Hardware implementations, on the other hand, can be faster, cheaper, and more compact (Togai and Watanabe, 1985; Yamakawa, 1988). The first hardware fuzzy chip was developed by M. Togai and H. Watanabe at AT&T Bell Laboratories using VLSI technology. T. Yamakawa pioneered fuzzy logic at the hardware level by developing a system that can accept linguistic information and perform approximate reasoning-based inference at very high speeds (more than one million Fuzzy Logic Inferences Per Second or FLIPS).

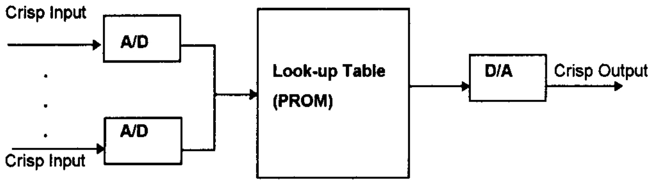

Figure 1. Block diagram of a PLD-based fuzzy logic controller.

This paper investigates a hardware implementation of a multi-input single-output fuzzy logic controller using a look-up table technique. This approach uses a programmable logic device. Speed is limited only by the A/D conversions, the PROM look-up table access, and the D/A conversion. Total conversion time is about 250 ns. This speed is more than 5 times faster than the design of the chips mentioned previously by M. Togai and H. Watanabe and T. Yamakawa, since the process requires only a look-up table.

PLD-BASED FUZZY LOGIC CONTROLLER

Like ASICs, PLDs can implement logic functions in a smaller area with fewer packages to obtain more compactness than standard small scale integration/medium scale integration (SSI/MSI) logic devices. Unlike ASICs, which require long design cycles and high development costs, PLDs use short design cycles and relatively inexpensive devices. With user programmable and configurable capabilities, programmable logic devices also allow design changes for more optimal solutions. This results in significant cost savings by reducing the risks of hardware design changes.

A hardware implementation of the PLD-based fuzzy logic controller can be accomplished by placing the results required by the fuzzy logic controller into a PROM. To obtain the look-up table for the PROM, the desired resolutions of all the input and output variables must be selected. Resolution is the number of subsegments that are divided along the input and output universes of discourse.

The result of $\log_2$ (resolution) gives the number of bits required for each input and output. After the input and output parameters have been determined, the look-up table is obtained via an off-line computer software program that performs the fuzzy inference process computations for all possible input combinations. The number of rows in the look-up table is equal to the product of the resolutions of the input universes of discourse. Figure 1 shows the overall block diagram of the PLD-based fuzzy logic controller. A/D converters are used to convert the crisp inputs into the desired input or "address" for the PROM while a D/A converter is used to convert the PROM's output into a control signal. Since all of the inference processes are performed off-line, the end result of this approach is a very high speed fuzzy controller. The classical backing truck to ramp example (Kong and Kosko, 1992; Kosko, 1992; Wilamowski and Sandige, 1993) is examined in this paper. The input fuzzy variables are **x** (distance) and ϕ (angle) of the truck's position. The output fuzzy variable is **e**, the error function. The values and the membership functions for

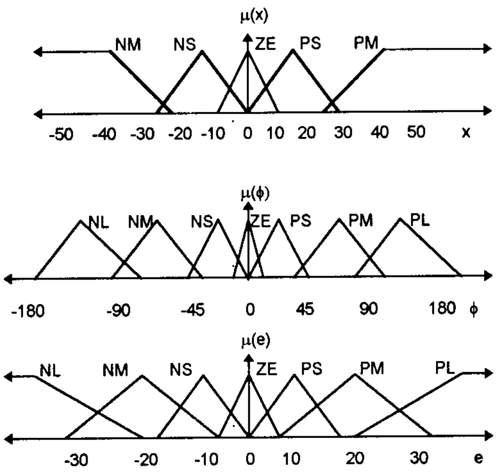

Figure 2. Fuzzy membership function used in experiments.

TABLE 1: FUZZY RULES USED IN THE SYSTEM

		x				
		NM	NS	ZE	PS	PM
	NL	PS	PM	PM	PL	PL
	NM	NS	PS	PM	PL	PL
	NS	NM	NS	PS	PM	PL
Φ	ZE	NM	NS	ZE	PS	PM
	PS	NL	NM	NS	PS	PM
	PM	NL	NL	NM	NS	PS
	PL	NL	NL	NM	NM	NS

All entries correspond to e

the inputs and output are given in Figure 2. Table 1 shows the fuzzy rules used in the system. A total of 35 (5 x 7) fuzzy rules are applied to the system.

The original control surface using Kong and Kosko's controller is shown in Figure 3 with the corresponding truck trace shown in Figure 4. The truck always starts with the same 35 positions corresponding to the center of the 35 fuzzy rules. In all of the following examples, 8 bits of output resolution are used for the D/A converter. A classical fuzzy controller was used to generate the contents of the PROM. First consider the very simple case of only 2 bits of resolution for each of the two inputs. In this case, a four-bit address is required and the total number of byte-size storage locations in the PROM (assuming 8 bits of output resolution for the D/A converter) is 2^4 = 16 locations. Using 2 bits of resolution for each input, the corresponding control surface is shown in Figure 5 and the truck trace is shown in Figure 6. The control surface and corresponding truck trace for 4 bits of resolution for each input are shown in Figure 7 and Figure 8, respectively. Finally, with 6 bits

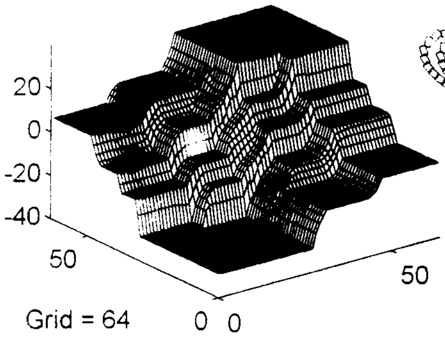

Figure 3. Control surface obtained with a classical fuzzy controller.

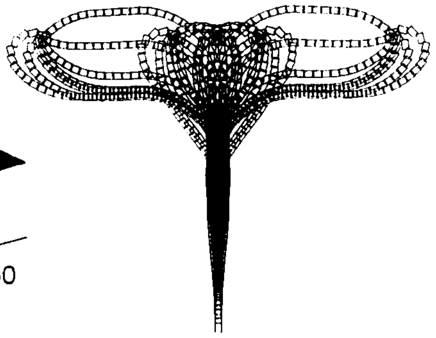

Figure 4. Trace of the truck using a classical fuzzy controller.

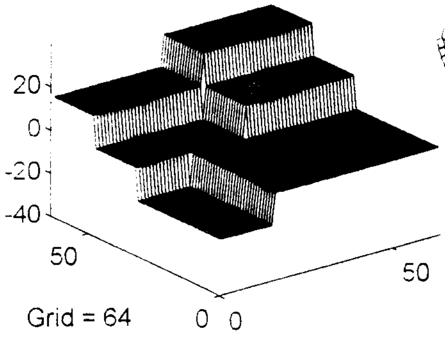

Figure 5. Control surface for 2 bits of resolution for each input.

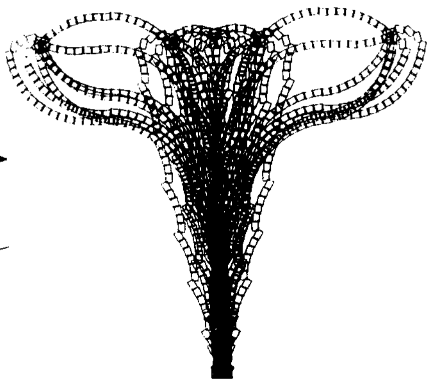

Figure 6. Trace of the truck using 2 bits of resolution for each input.

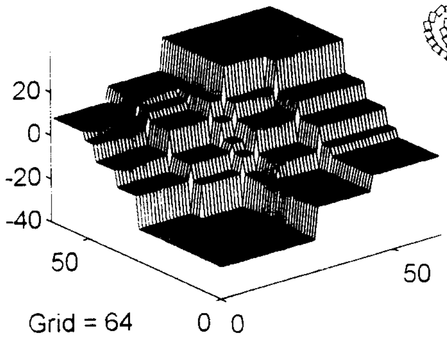

Figure 7. Control surface for 4 bits of resolution for each input.

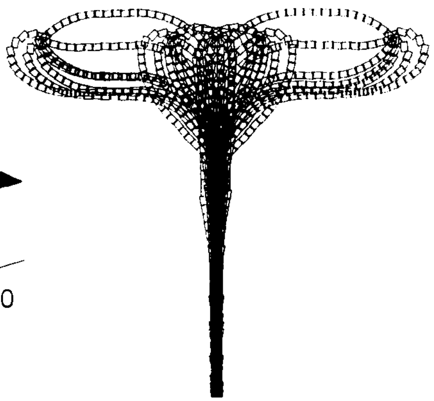

Figure 8. Trace of the truck using 4 bits of resolution for each input.

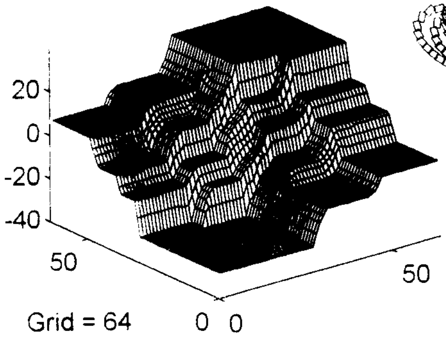

Figure 9. Control surface for 6 bits of resolution for each input.

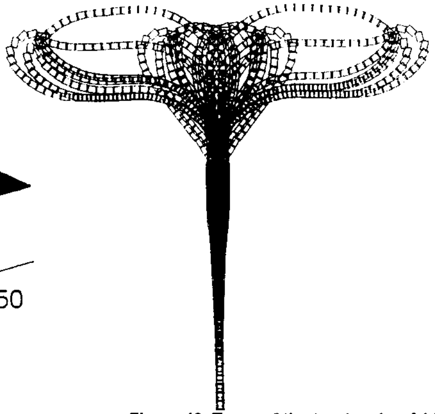

Figure 10. Trace of the truck using 6 bits of resolution for each input.

of resolution for each input, the control surface and the corresponding truck trace are shown in Figure 9 and Figure 10, respectively.

As one can see from Figures 5 to 10, 4 bits of resolution for each input provide reasonable accuracy. In this case, the required PROM storage locations are $2^8 = 64$ byte size locations, and this is a relatively small PROM. One can see that it is practical to increase the number of inputs to 4. For the case of 4 bits of resolution for each input, the PROM address is 16 bits which corresponds to $2^{16} = 64$ K byte-size storage locations.

CONCLUSION

A hardware implementation technique for a PLD-based fuzzy logic controller has been presented. Simulation results show that the fuzzy logic's interpolation capabilities work well with a fairly low number of bits, that is, 4 to 6 bits. The technique present is very cheap (cost of the A/D converters, the PROM, and the D/A converter). In addition, these types of controllers are very fast, with a conversion time of only about 250 ns.

REFERENCES

Togai, M., Watanabe, H., (1985). "A VLSI implementation of a fuzzy inference engine: Toward an expert system on a chip," Proc. of 2nd Int. Conf. on AI and Applications, Dec, 192-197.

Yamakawa, T., (1988). "High-Speed Fuzzy Controller Hardware Systems: The Mega FLIPS Machine," Information Science, Elsevier Science Publishing Company, Inc., 113-128.

Kong, S., Kosko, B., (1992). "Adaptive fuzzy system for backing up a truck-and-trailer," IEEE Trans. on Neural Networks, (3), March, 211-223.

Kosko, B., (1992). Neural Networks and Fuzzy Systems - A Dynamical Systems Approach to Machine Intelligence, Prentice Hall.

Wilamowski, B. M., Sandige, R. S., (1993). "Trainable Fuzzy Controller," Intelligent Engineering System Through Artificial Neural Networks, ASME Press, (3), 561-566.

INTELLIGENT ENGINEERING SYSTEMS THROUGH ARTIFICIAL NEURAL NETWORKS

VOLUME 4

Editors:

Cihan H. Dagli
Benito R. Fernández
Joydeep Ghosh
R. T. Soundar Kumara