A Field Programmable Analog Array for Continuous, Fuzzy, and Multi-Valued Logic Applications[1]

Edmund Pierzchala, Student Member, IEEE †‡
Marek A. Perkowski, Member, IEEE †‡
Stanislaw Grygiel ‡

†Analogix Corporation, tel: (503) 653–6827
‡Department of Electrical Engineering, Portland State University
P.O. Box 751, Portland, Oregon, 97207–0751
tel: (503) 725–3823, e–mail: edmundp@ee.pdx.edu

Abstract

In this paper we propose a novel approach to the realization of continuous, fuzzy, and multi-valued logic (mvl) circuits. We demonstrate how a general-purpose field programmable analog array (FPAA), with cells realizing simple arithmetic operations on signals, can be used for this purpose. The FPAA, which is being implemented in a bipolar transistor array technology, operates from $\pm$3.3V or $\pm$5V power supplies and works in the range of frequencies up to several hundred MHz.

1. Introduction

Digital programmable devices for classical, two-valued logic have grown in importance over recent years. Although there are many published circuits for multiple-valued logic (e.g. [8, 18, 19]), as well as analog logic (including continuous logics, fuzzy logic and Lukasiewicz logic as particular examples) [9, 10], to the best of the Authors' knowledge there are no programmable devices for those logics. In this paper we show how a general-purpose programmable analog array (FPAA) presented in [15], can be used for the implementation of a wide class of mvl, fuzzy logic, and continuous logics circuits. Minor modifications extending the set of nonlinear operations realized by individual cells of the device are applied in order to reduce the number of cells necessary to realize basic logic operations.

Both the structure of the device and the functionality of individual cells were chosen primarily for dynamic system type of applications, but they prove to be well suited for the realization of various mentioned logics. The presented examples are linked to the theory of realizing mvl functions in finite fields, introduced in [12, 13], and generalized for mvl in [14].

2. The device

The FPAA is based on a regular, square array of current-mode processing cells, interconnected on two levels: local, and global. Figures 1A and B show the local and the global

1. Patent pending.

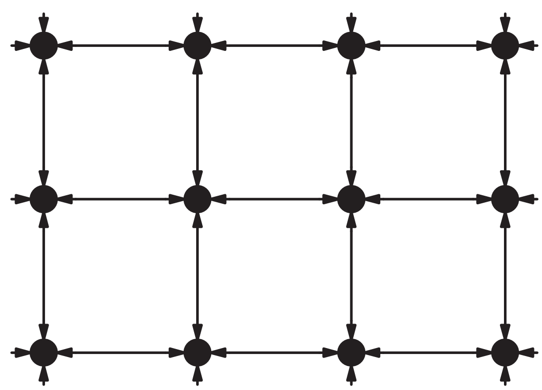

FIGURE 1A. Local signal interconnections of the FPAA.

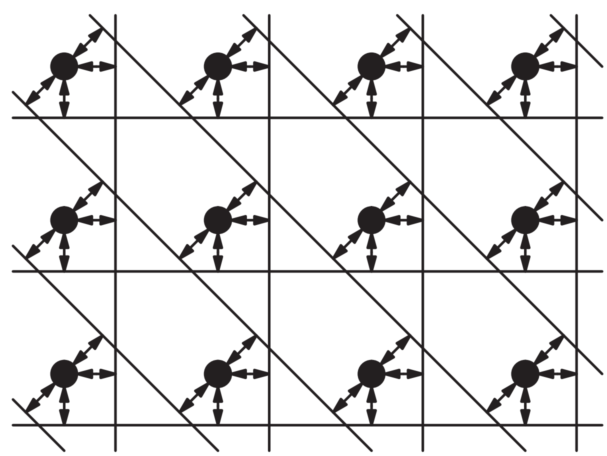

FIGURE 1B. Global signal interconnections of the FPAA.

interconnection patterns, respectively. Each cell is connected to its four nearest neighbors by a two-way current-mode signal interconnection, and thus is able to receive four different signals produced by those neighbors, whether all of them, or just selected ones. The cell's own output signal is programmably distributed to the same four neighbors (Figure 1A). The global interconnection pattern is superimposed on the local one, but it is shown separately to avoid clutter (Figure 1B). Each cell can broadcast its current-mode output signal independently to any of the four global lines to which the cell is connected. Signals sent to a global line from different cells are summed on this line. If more than one cell receives the signal from a given global line, the signal is divided evenly by the receiving cells. Each cell can then send and receive signals to and from any of its four nearest neighbors and any of the four global lines to which it is connected. Each cell thus has eight inputs and one output (in fact, eight outputs with copies of the same output signal).

A functional diagram of the cell is shown in Figure 2. The cell processes current-mode, differential signals. Due to the lack of space only a general description of the cell's analog signal processing circuitry will be given. Detailed design of the cell will be presented in [16].

Figure 3 shows the main building block of the cell [2, 4, 5, 6]. In its simplest form the circuit contains only transistors $Q_1 - Q_4$ and current source I_B^+. Current sources I_A represent the circuit's input signals. The circuit is fully differential, i.e. both input and output signals are represented by differences of currents in two wires. The sum of currents

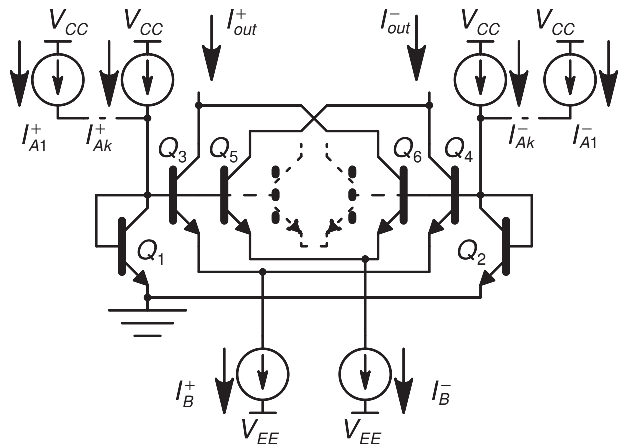

FIGURE 3. Basic building block of the cell.

I_A^+ which can be expressed as $I_A(1 + X)$ is the positive "half" of the input signal, and I_A^-, which can be expressed as $I_A(1 - X)$, is its negative "half". The input signal is then $I_A(1 + X) - I_A(1 - X) = 2I_AX$; X is called *modulation index*. Likewise, the output signal is the difference $I_{out}^+ - I_{out}^-$, expressed as $I_B(1 + Y) - I_B(1 - Y) = 2I_BY$. Current gain is determined by the ratio I_B/I_A and in practice can be tuned over several decades from a fraction of unity to about 10. The circuit has an excellent linearity and a wide bandwidth, limited by the f_T of the transistors [2]. In the bipolar process used for prototyping f_T is of the order of 8 GHz and the simulated –3dB bandwidth of this circuit

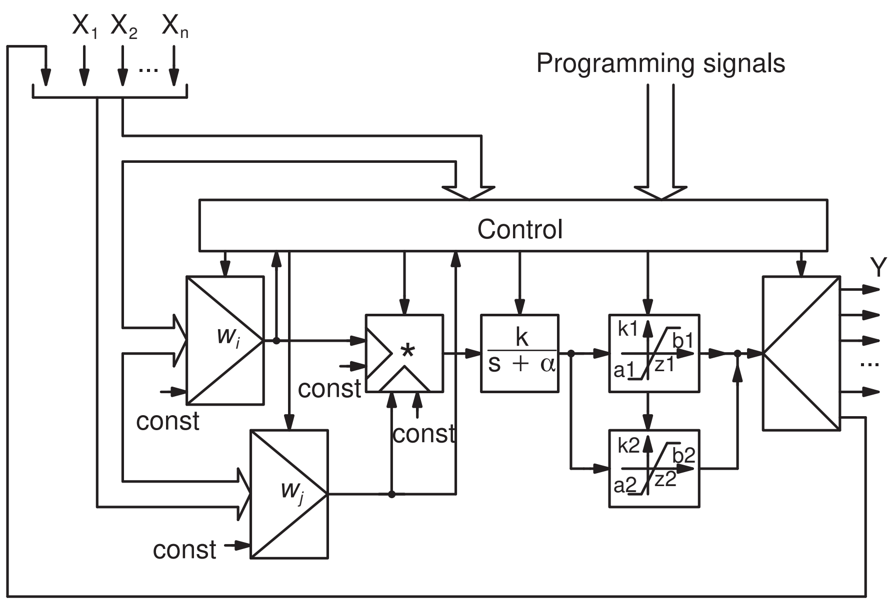

FIGURE 2. Functional diagram of the programmable cell.

is over 3 GHz for I_B of several hundred μA. Figure 4A

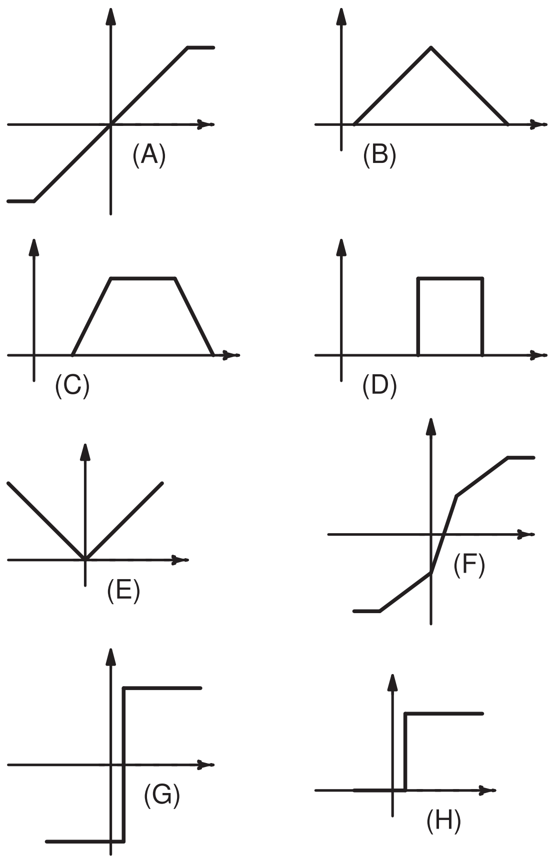

FIGURE 4. Selected DC transfer characteristics of the cell.

shows the DC transfer characteristic of the circuit. The slope in the linear range can be changed by adjusting the gain. The width and height of the linear range are determined by the currents I_A and I_B, respectively. By adding (subtracting) currents on the input and on the output of the circuit (by additional programmed current sources) one can change the location of the zero of the characteristic, as well as the two clipping (saturation) levels.

This circuit has countless variations. By including transistors Q_5 and Q_6 one achieves the ability to invert the signal (negative weight). If another pair of inputs is connected in place of the tail current sources I_B^+ and I_B^-, a current-mode Gilbert multiplier [3] is realized. More transistor pairs can be added (dashed line) to obtain more independently tuned outputs.

Figure 5 shows the schematic of a multiplexer-summer with independent tuning of input weights. Additional summation (without independent tuning) can be realized by connecting a number of signals to each input.

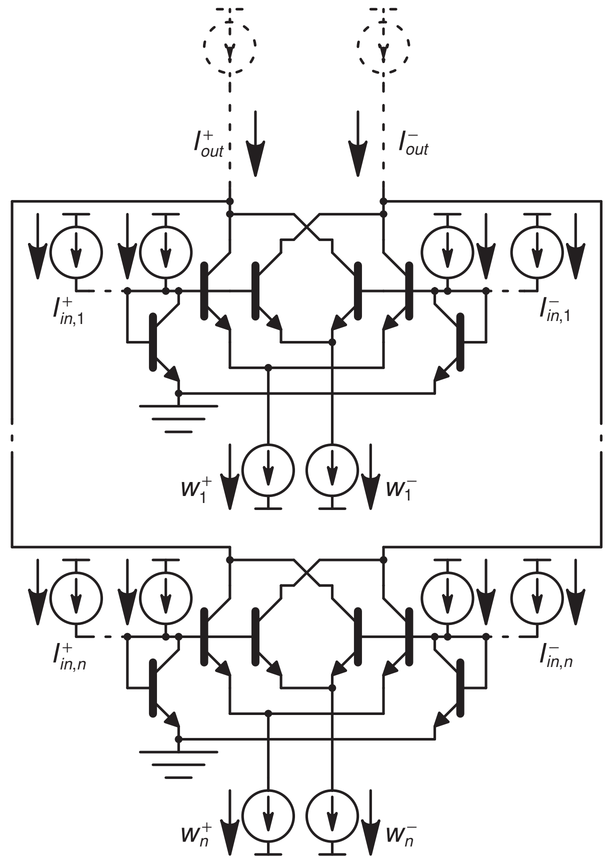

FIGURE 5. Multiplexer-summer.

A demultiplexer can be realized in a similar fashion by placing more inner (output) pairs of transistors. Circuits from this family can be connected in cascades by adding current sources (dashed line in Figure 5). Then the difference between the (constant) current from such sources and the output signal of one stage can be fed to the next stage. This arrangement has better frequency response than a pnp current mirror. By cascading several stages based on the circuit of Figure 3 a wide-band current amplifier tunable in the range of 0 – 80dB or more [2, 4] can be obtained. Two (or more) nonlinear blocks shown in Figure 2 can be realized as single amplifier stages each. With two blocks one can achieve many nonlinear characteristics, some of which are shown in Figures 4A–H.

The integrator (Figure 6) is realized by connecting an operational transconductance amplifier (OTA) input stage to a current amplifier. The current amplifier has an additional voltage-mode output. Capacitors C are connected to this output and to the input of the OTA, realizing a Miller integrator. The current-mode output signal of the amplifier is proportional to its voltage-mode output signal, which represents the integral of the input current-mode signal. In

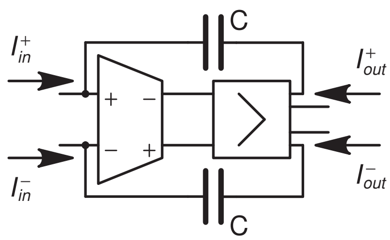

FIGURE 6. Current-mode integrator and sample-and-hold circuit.

this feedback arrangement the OTA works with a very small input voltage swing (provided that the gain in the loop is high); also, due to the feedback operation the voltage on the capacitors is only slightly disturbed by the nonlinearities within the loop. Thus the linearity of the circuit is primarily determined by the linearity of the relationship between the voltage-mode and the current-mode output signals, which is fairly good if the output stage is designed properly. The OTA input stage linearity is not critical. This design inherits all good features of the classical Miller integrator employing a voltage-to-voltage amplifier (an op amp), that is the ability to realize a low-frequency pole (ideally, an integrator's pole should be at zero) with small capacitors value, practically independent of the impedances of the source of the input signal and the load. This is because the capacitors see an extremely high impedance (typically of the order of tens or even hundreds of GΩ). In the traditional design of a current-to-current integrator, the Miller integrator (or even a capacitor) is followed by an OTA, converting the full range of voltages developing across the capacitor into the output current. In such a design the linearity of the OTA limits the linearity of the integrator, even though (in the Miller integrator) the voltage on the capacitors is a nearly perfectly linear integral of the input signal. The circuit has additional advantages over the classical design. The input signal can be fed directly into the current amplifier, making the voltage on the capacitors track the input signal. When desired, the input stage of the current amplifier can be turned off, and the capacitors will hold the last value of the signal. Finally, when no integration or sample-and-hold operation is necessary, the voltage output is turned off and only the current amplifier is used.

With C of 0.8pF the circuit demonstrates simulated phase response of $-90\pm0.5°$ in the range of several hundreds Hz to over 300 MHz. The pole can be moved by changing the operating conditions of the circuit. If a high frequency pole is desired, the output signal can be fed back to an additional input of the OTA to simulate resistors connected to the output. The integrator contains the only high impedance points in the circuit.

Different functions of the cell are programmed by changing the tail currents in the amplifier, multiplier, and integrator circuits. For instance, if no multiplication is required, a constant (tunable) current is fed into one input of the multiplier. This represents selection symbolically shown as *const* in Figure 2. In that case, the multiplexer-summer disconnected from the multiplier can still be used for the calculation of the signal connected to the control block.

Power dissipated by the circuit depends on the number of cells used, number of active inputs and outputs in each cell, tail currents, and power supply voltages. For $\pm$3.3V supply voltages, four inputs active, and bias currents of 200μA, power dissipated by a single cell is about 40mW.

The *control block* stores programming information received from the outside of the cell, and sends control signals to analog processing blocks of the cell. Additionally, it performs comparison of the input signals against signals produced by the analog processing blocks or a programmed constant (Figure 7). Each of the comparators produces two

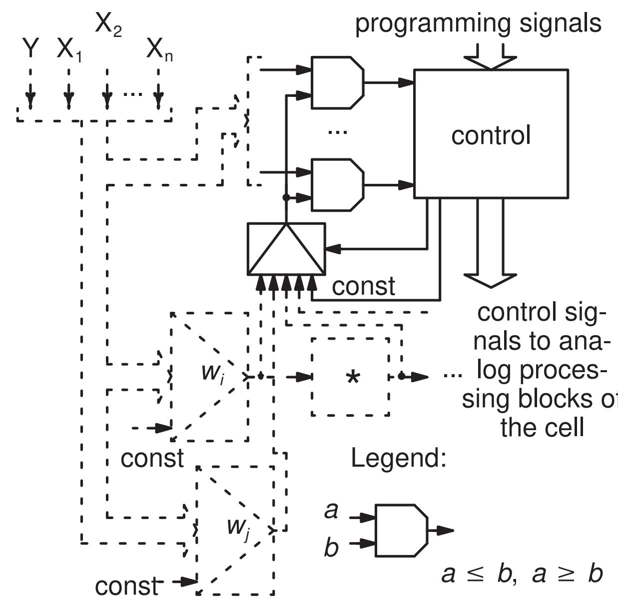

FIGURE 7. Control block of the cell.

binary signals corresponding to the conditions $a \leq b$ and $a \geq b$. Two signals of equal value on the output of any comparator indicate equal input signals. This way the control block can produce control signals being a function of certain conditions of instantaneous input signal values, such as equality of two or more signals, equality of a number of signals to zero or another constant, etc. This feature is also used to realize *minimum and maximum follower (min* and *max)*, *absolute value* (*abs*), and other operations. To realize *min* and *max* operations the control block simply detects the smallest (largest) signal and selects this signal on the input of the cell. This is accomplished by comparing the output signal of the selected multiplexer-summer with the input signals. If one (or more) of the input signals are smaller (larger) then the multiplexer-summer output, the control circuit sends appropriate signals to the multiplexer-summer to adjust its weights until the smallest (largest) signal is selected. When realizing the *abs* function, the control block changes the sign of input weights if the weighted sum is negative.

Some operations important for mvl, fuzzy and other logic applications, performed by the cell, are summarized in Table 1. X_i denote input signals, Y denotes the output signal. No distinction is made between local and global signals, since the cell processes them in the same manner.

Table 1.

1. $Y = k\,(\sum_{w_i \in W_1} w_i X_i)\,(\sum_{w_j \in W_2} w_j X_j)$

   W_1 and W_2 are independent sets of input weights; k is tuned in the range 0 – 80dB. Complements of the signals (to the maximum possible signal value, Max) can be calculated.

2. $Y = k\,(\sum_{i \in W} w_i X_i)$
3. $Y = k\,X_i\,X_j$
4. $Y = k\,X_i^2$
5. $Y = k\,\min(X_1, ..., X_n)$ — The control block “watches” input signals and selects the smallest one.
6. $Y = k\,\max(X_1, ..., X_n)$
7. $Y = k\,Y_{1-6}\frac{1}{s+\alpha}$ — Y_{1-6} is any of the functions presented in rows 1–6 above; $\alpha \geq 0$.
8. $Y = a\,sign(Y_{1-6})$ — $a = b$, $k = \infty$.
9. $Y = b\,U(Y_{1-6})$ — U denotes the step function. $a = 0$, $b = Max$, $k = \infty$.
10. $Y = k\,|Y_{1-6}|$
11. $Y = X_i$ — Identity.

As seen from the table, the cell performs summing of input signals selected by the control circuitry, multiplication of two signals (squaring of one signal), or multiplication of two independently derived weighted sums of input signals. Further processing includes lossless or lossy integration, and clipping.

The functions shown above are important for the implementation of continuous-time dynamic systems [7], and multi-valued, fuzzy, and continuous (such as Lukasiewicz) logic circuits.

The architecture of the device is motivated by the desire to enable circuit realizations with minimal signal delays. A number of examples, including an elliptic eighth-order ladder filter [17], a rank filter cell [11], circuits for tracking the product of two matrices, a solution of a system of linear equations [7], and a solution of a linear programming problem by the method of steepest descent [7], are shown in [15].

The cell is being implemented in a bipolar transistor array process. An implementation in BiCMOS and analog CMOS (to increase density) is considered.

3. MVL applications

3.1. Galois field 2^2 operations

Figures 8A and B show the tables for addition and multi-

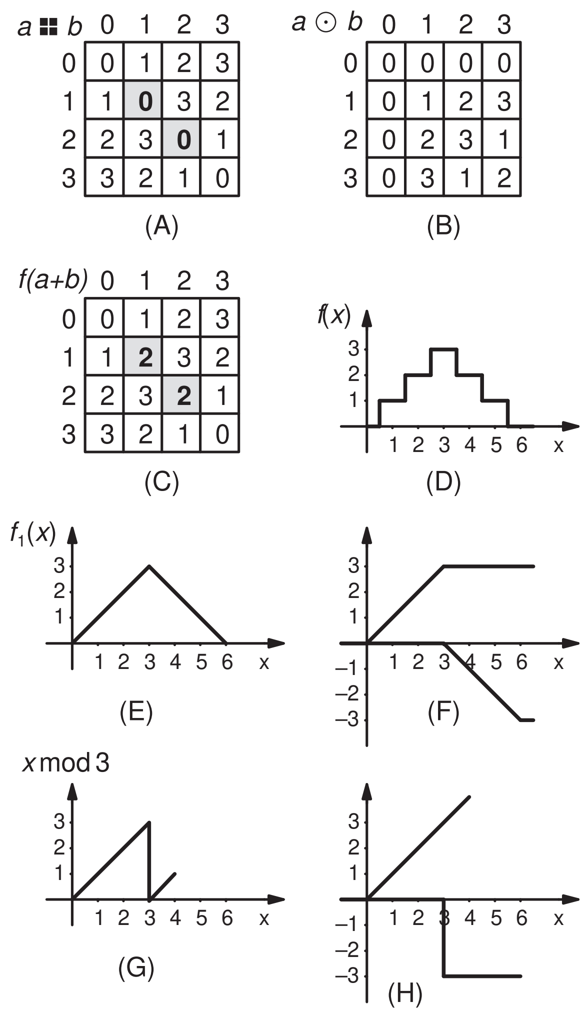

FIGURE 8. GF(2^2) operations.

plication in Galois field of four elements. Each of these operations can be realized by a single cell of FPAA, assuming that only two of the cell’s inputs are used at a time. Addition can be realized as $a \boxplus b = f(a+b)$ for $a \neq b$ (Figures 8C and D), and $a \boxplus b = 0$ otherwise. The condition $a=b$ can be detected by the control block. This requires programming the weights of one of the input multiplexers-summers to calculate the difference $a-b$ of the input signals, selecting constant 0 for comparison in the control block, and controlling the weights of the other input multiplexer to set them to zero if $a=b$ was detected. Instead of function $f(x)$ (Figure 8D) a smooth function $f_1(x)$ (Figure 8E) can be used. This function can be realized by adding two characteristics of the saturation blocks shown in Figure 8F. If the function of the form shown in Figure 8D is required, it can be realized by providing more nonlinear blocks in the cell.

Multiplication $a \odot b$ in the field (Figure 8B) can be realized as $a \odot b = ((a+b-2) \bmod 3) + 1$ for $a \neq 0$ and $b \neq 0$, and $a \odot b = 0$ otherwise. The two conditions for a and b can be tested independently by the comparators in the control block, and upon at least one of them being true the input weights of the multiplexer-summer would be turned down to 0. *Mod* 3 operation can be realized as shown in Figures 8G and H. The control block performs the necessary logic operations.

The realizations of $GF(2^2)$ operations proposed above are similar to the ones presented in [19].

3.2. Orthogonal expansion structures

Having defined the addition and multiplication in $GF(2^2)$, we can apply the combinational functions synthesis method based on orthogonal functions presented in [14]. Figure 9A shows a block diagram of a structure realizing a function of input variables $X_1, X_2, ..., X_m$. Each column realizes one orthogonal function over $GF(2^2)$. Multiplied by a constant from $GF(2^2)$, this function is added to the other orthogonal functions. All operations are in $GF(2^2)$. Figure 9B shows an example of realization of one of the functions f_i. Since each cell can realize the identity operation (see Table 1), it is possible to omit certain input variables X_i, X_3 in this example. More than one column of cells can be used for the realization of each f_i if necessary. Also, it may be convenient to make certain input variables available on more than one horizontal line. An alternative approach, based on providing literals on horizontal lines, or some functions of single variables which are convenient for the creation of literals, is also possible. In one such approach the powers (i.e. multiple products in $GF(2^2)$) would be used to create polynomial expansions of mvl functions.

3.3. Post logic

Same structures shown in Figure 9 can be used for the implementation of Post logic. Each cell can realize *min* and *max* operations (Table 1), and literals of the form shown in Figure 4C. Each function f_i is realized as in Figure 9B, except that the cells realize *min* or identity operation. Instead of summing over $GF(2^2)$, *max* operation is used.

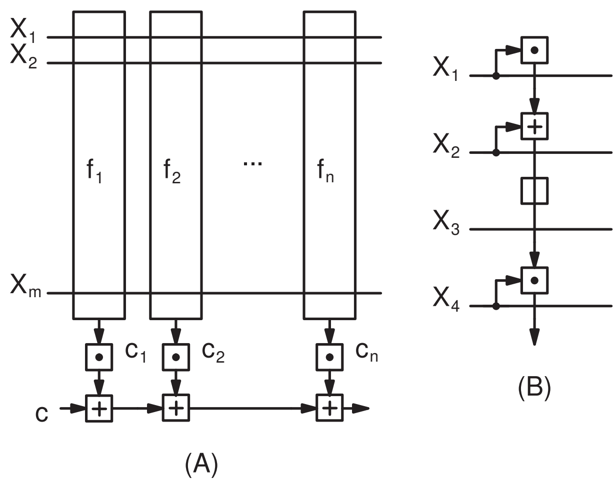

FIGURE 9. Orthogonal expansion structures.

3.4. Other logics

The structure of Figure 9A can be used for realization of combinational functions with other methods. Such realizations, unlike the ones based on the orthogonal expansions, may not be unique in the presented structure, however due to the availability of addition, multiplication (in the conventional sense), and nonlinear operations on signals, some combinational functions may have very efficient implementations.

Also, the topology of mvl circuits mapped into the FPAA does not have to be constrained to the form shown in Figure 9. Global vertical and diagonal signal lines can be used, if necessary, to achieve greater flexibility of the circuits' topologies. Figure 10 shows a structure for implementations based on generalized Shannon expansion of mvl functions [14]. Some input variables need to be connected to more than one diagonal line. More general forms of the same kind are possible, based on other operators than > used for separation, for instance even vs. odd parity, based on matrix orthogonality [14], which is a generalization of the approach presented in [12, 13] for two-valued functions.

Finally, the integrator block can be used as a memory element, enabling realization of sequential circuits. Since each cell can realize identity function, and global connections are available, larger irregular structures, composed of combinational and sequential parts can be built in the presented structure.

4. Analog logics

Fuzzy logic and continuous logics (such as Lukasiewicz logic) [9, 10] can be realized as well. As an example, let us consider an implementation of a fuzzy logic controller

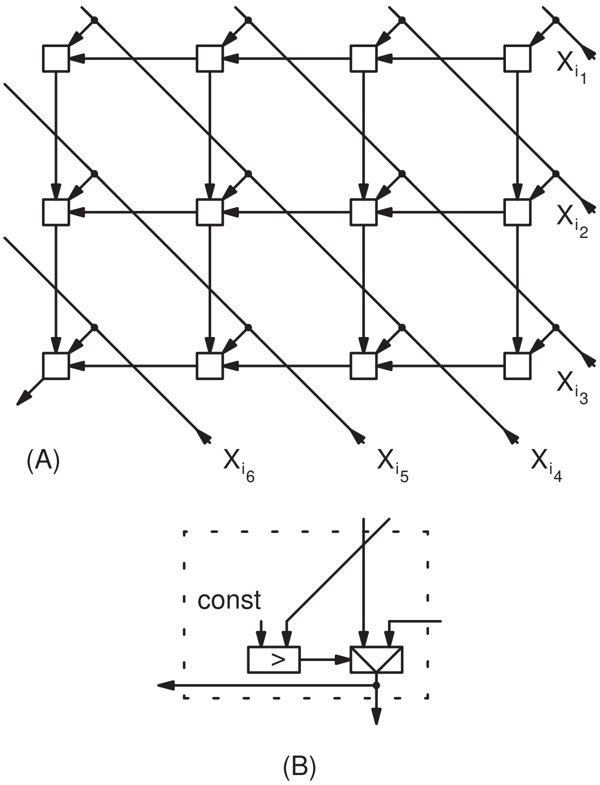

FIGURE 10. Generalized Shannon expansion structure.

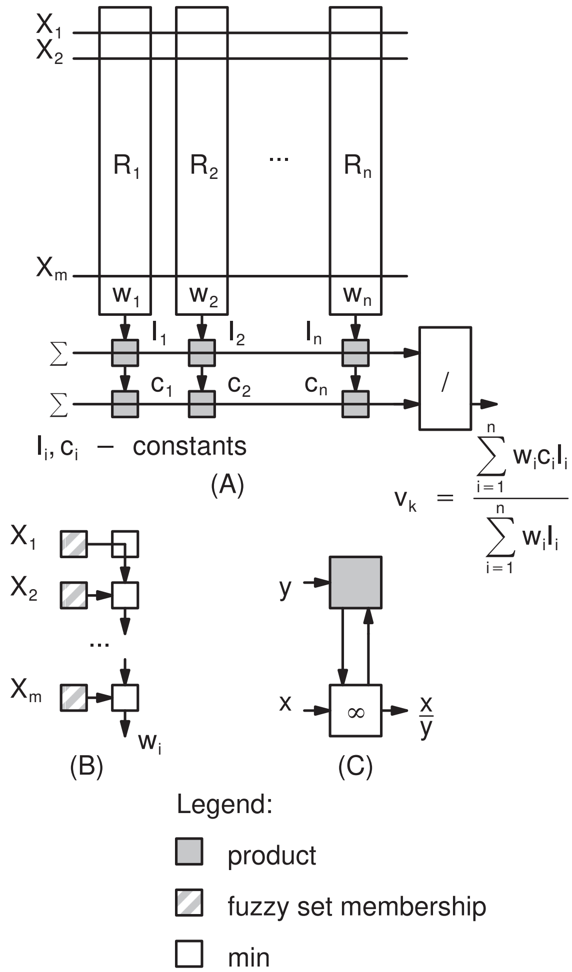

FIGURE 11. Fuzzy controller.

with correlation-product inference [8]. A structure very similar to that of Figure 9A, shown in Figure 11A, is used to implement a controller with m input variables and n fuzzy inference rules. Figure 11B shows details of each rule implementation. Fuzzy membership function is implemented as a trapezoidal transfer function of the kind shown in Figure 4C. Activation values w_i are multiplied by centroid values of the fuzzy rules consequents c_i, and their areas I_i, yielding two sums computed on two horizontal global lines. The final expression for the defuzzified output variable v_k is produced by a two-quadrant divider [7] shown in Figure 11C.

5. Conclusions

In this paper, we have demonstrated how a general purpose field programmable analog array can be used for the implementation of various logic circuits. Although the FPAA was designed for dynamic systems and general electronics, its structure and functionality is generally well suited to the presented applications. Only minor modifications of the FPAA introduced in [15], namely the inclusion of two or more nonlinear blocks, are necessary to enable convenient realization of the presented examples. It is demonstrated that the realizations based on orthogonal expansions as well as more general ones, based on sets of not necessarily orthogonal functions, lead to regular circuit structures which can be easily mapped to the FPAA.

The FPAA is an excellent tool for fast prototyping of circuits in various logics. It will provide the researchers in the field with an opportunity to experiment with hardware realizations of various logic circuits without the necessity to design and fabricate them. Presented examples demonstrate simplicity of realization of a wide class of such circuits, which will also enable the implementation of design automation procedures.

The FPAA is under implementation in a bipolar transistor array technology. For future applications, a programmable device dedicated to mvl, fuzzy and other logics can be implemented based on the presented material.

6. References

1. K. Current, “Multiple Valued Logic: Current-mode CMOS Circuits”, pp. 176–181, *ISMVL’93*.
2. Gilbert, B., “A New Wide-Band Amplifier Technique”, *IEEE Journal of Solid State Circuits*, Vol. SC–3, No. 4, pp. 353–365, Dec. 1968.
3. Gilbert, B., “A Precise Four-Quadrant Multiplier with Subnanosecond Response”, *IEEE Journal of Solid State Circuits*, Vol. SC–3, No. 4, pp. 365–373, Dec 1968.
4. Gilbert, B., “Current-mode Circuits From a Translinear Viewpoint: A Tutorial”, in: *Analogue IC Design: the current-mode approach*, ed. C. Toumazou, F. J. Lidgey, D. G. Haigh, pp. 11–91, Peter Peregrinus Ltd., 1990.
5. Grebene, A. B., *Bipolar and MOS Analog Integrated Circuit Design*, J. Wiley, 1984.
6. Grey, P. R., and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits*, 3rd ed., J. Wiley, 1993.
7. Hausner, A., *Analog and Analog/Hybrid Computer Programming*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1971.
8. B. Kosko, *Neural Networks and Fuzzy Systems, A Dynamical Systems Approach to Machine Intelligence*, Prentice Hall, Englewood Cliffs, NJ, 1992.
9. J. W. Mills, “Area-Efficient Implication Circuits for Very Dense Lukasiewicz Logic Circuits”, *ISMVL ’92*, pp. 291–298, May 1992.
10. J. W. Mills, “Lukasiewicz’ Insect: The Role of Continuous-Valued Logic in a Mobile Robot’s Sensors, Control, and Locomotion”, *ISMVL’93,* pp. 258–263.
11. Paul, S., Hümper, K., and J. A. Nossek, “A Simple Analog Rank Filter”, *ISCAS,* pp. 121–124, San Diego, CA, 1992.
12. M. A. Perkowski, “A Fundamental Theorem for EXOR Circuits”, *Proc. IFIP W.G. 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pp. 52–60, Hamburg, Germany, Sep 1993.
13. Perkowski, M. A., Sarabi, A., and F. R. Beyl, “Universal XOR Canonical Forms of Switching Functions”, *Proc. IFIP W.G. 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pp. 27–32, Hamburg, Germany, Sep 1993.
14. Perkowski, M. A., and E. Pierzchala, “New Canonical Forms for Four-valued Logic”, Internal Report, Department of Electrical Engineering, Portland State University.
15. Pierzchala, E., and M. A. Perkowski, “High Speed Field Programmable Analog Array Architecture Design”, *FPGA ’94*, Berkeley, California, Feb 1994, available from the authors.
16. E. Pierzchala, “The Design of a Current-Mode Signal Processing Cell for a Field Programmable Analog Array”, in preparation.
17. Tan, M. A., *Design and Automatic Tuning of Fully Integrated, Transconductance-Grounded Capacitor Filters*, Ph.D. Thesis, Univ. of Minnesota, 1988.
18. Taniguchi, K., Sasaki, M., Ogata, Y., Ueno, F., and T. Inoue, “BiCMOS Current Mode Multiple Valued Logic Circuits with 1.5V Supply Voltage”, *ISMVL’92*, pp. 216–220.
19. Zilic, Z., and Z. Vranesic, “Current-mode CMOS Galois Field Circuits”, *ISMVL ’93,* p. 245–250.