VLSI IMPLEMENTATION OF CELLULAR AUTOMATA : MODULO-ARITHMETIC UNITS

T.A. York*, B. Srisuchinwong*, P.J. Hicks*, Ph. Tsalides** and A. Thanailakis**

Cellular Automata (CA) and their natural affinity for VLSI are discussed. Group behaviour of CA is related to that for modulo-arithmetic units and this leads to an architecture for modulo multiplication. Implementation of a mod-127 multiplier using two techniques, full custom CMOS design and Xilinx programmable gate arrays is described. The principle of operation is based on two identical 12x12, null-bounded, CA each having semi-group order 126. In practice, implementation utilises the data compression capabilities to reduce the area required by these arrays by about 90%. This is achieved by using triangular CA, each comprising 15 cells that have appropriately chosen initial boundary conditions. Encoding and decoding is performed on-chip and the complexity of this task is significantly reduced by observing only critical cells. Performance and area requirements for this modest function are revealed such that predictions can then be made for more substantial units.

1. Introduction

The theoretical foundations of modern Residue Number System (RNS) theory were developed in the eighteenth and nineteenth centuries by Euler, Fermat, Gauss and other notable mathematicians. The major factor that has influenced the revival of RNS arithmetic was the emergence, by the mid-1960's, of digital signal processing as a technical subject distinct from general digital computing.

When the VLSI era arrived in the late 1970's, it was found that traditional algorithms used in digital signal processing did not "parallelise" nor "modularise" easily, both of which are essential to fully utilise VLSI technology. It became clear that new design approaches were needed that could incorporate modularity, parallelism and fault tolerance. It was noted by several researchers that such features can be provided by residue arithmetic (Soderstrand, Jenkins, Jullien and Taylor, 1986), due to its modular algebraic structure. The potential of residue arithmetic for realising high speed signal processing has been demonstrated in many application areas involving digital filters and transforms (Soderstrand, Jenkins, Jullien and Taylor, 1986; Jenkins and Leon, 1977; Soderstrand, 1977; Tseng, Jullien and Miller, 1979). The common feature that emerges from this earlier work is the use of mod-p multipliers as fundamental building blocks. Such multipliers can be implemented by several techniques, such as direct lookup tables using semiconductor memory, index calculus (Fraenkel, 1961;

* Dept. of Electrical Engineering and Electronics, UMIST, Manchester M60 1QD, UK.
** School of Engineering, Democritus University of Thrace, 67100 Xanthi, Greece.

Soderstrand, 1983), square-law multipliers (Soderstrand and Vernia, 1980), and others (Yau and Chung, 1976; Taylor, 1982; Taylor and Huang, 1982). Recently, new techniques for implementing mod-p multipliers using cellular automata (CA) have been reported (Pries, Thanailakis and Card, 1986; Tsalides, Thanailakis, Card and Pries, 1987).

Cellular automata have been of theoretical interest since the pioneering work of von Neumann (Neumann, 1966) and others in the 1940's. Briefly, they consist of a regular uniform lattice of identical cells in an N-dimensional space each communicating with a specific set of its neighbours. Each cell is capable of existing in a finite discrete state space where, in general, the state space is quite small, typically 1 or 0. A CA is characterised by four basic properties : the cell geometry, the neighbourhood specification, the number of states per cell and finally the localised algorithm of computation. Unlike the related systolic arrays the cells in a CA perform only basic computations, typically exclusive-or. Due to the mesh connection of the cells in a physical layout, the CA will exhibit minimal wire area and therefore such structures are particularly well matched to implementation using VLSI.

A new technique for the design and VLSI implementation of a mod-127 multiplier based on 2-D CA is presented in this paper. This technique utilises the data compression capabilities of CA to reduce the silicon area required for these units by about 90%, by exploiting the symmetry existing in the states of CA. To minimise the demands on silicon fabrication for this investigative project the present device has been restricted to a, modest, mod-127 multiplier. This is sufficient to satisfy the objective, namely to reveal complexity measures for the approach such that, by extrapolation, its suitability for larger, more useful, multipliers can be evaluated relative to alternative techniques.

This paper describes the basic features of cellular automata and their application to modulo-arithmetic units. Two implementations of a mod-127 multiplier are described, in full custom CMOS and as a Xilinx programmable gate array, and these are compared to other approaches in terms of size and performance.

2. Cellular Automata

Cellular automata comprise simple, identical, cells arranged in a regular array. The cells can be arranged in one (Nx1), two (NxN) or three dimensions (NxNxN). Each cell communicates with those cells in its immediate vicinity, the neighbourhood, leading to a highly desirable situation when considering silicon implementations, namely a preponderance of local interconnections. The state of each cell is updated, synchronously, at each clock cycle according to a local rule, typically exclusive-OR. At the extremities of the array CA exhibit either null or periodic boundaries. Each arrangement of local states for the cells corresponds to a global state for the CA. The present work is restricted to arrays in which the neighbourhood is composed of only the nearest neighbours and information is not saved beyond the previous iteration.

Examples of simple one-dimensional CA are shown in Figures 1, 2 and 3 below. The local rule of operation is exclusive-OR in each case and either null or periodic

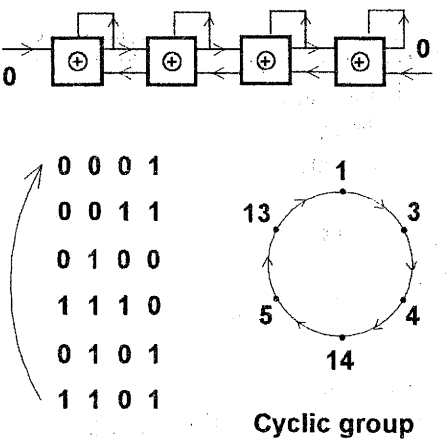

Fig. 1. Rule 150 - Null boundary.

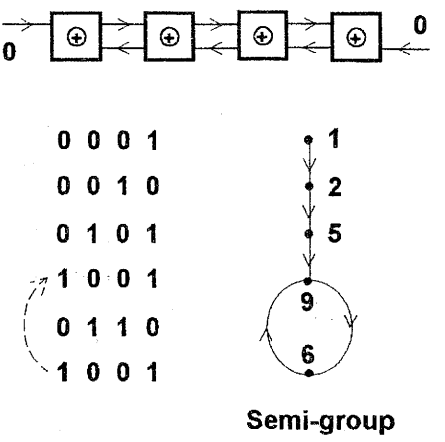

Fig. 2. Rule 90 -Null Boundary.

boundaries are used. In Figures 1, 2 and 3 the sequence of binary words, corresponding to iterations of the global state of each CA are shown on the left hand side of each figure. As can be seen on the right-hand side of each figure CA can exhibit a variety of characteristics. Figure 1 shows a cyclic group, of order 6, with each state having unique predecessor and successor states. Figure 2 displays semi-group behaviour. Each state has a unique successor but there are a variety of combinations of predecessors. State '1' is a so-called 'garden of eden' state, having no predecessor, states 2, 5 and 6 have unique predecessors and state 9 has two possible predecessors. In contrast, Figure 3 displays no group behaviour. There are many 'garden of eden' states with no predecessors and one 'graveyard' state whose only sucessor is itself.

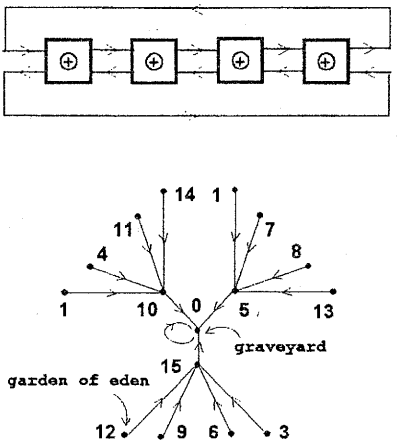

Fig. 3. Rule 90 - Periodic Boundary.

For 1-D CA the local rules associated with each cell are commonly referred to using names that were introduced by Wolfram (1983). These names are derived from operations on the three-cell neighbourhood as shown in the examples below :

Rule 150 : Exclusive-OR of a cell and its two neighbours :

111	110	101	100	011	010	001	000
1	0	0	1	0	1	1	0

binary 10010110 = decimal 150

Rule 90 : Exclusive-OR of the two neighbours :

111	110	101	100	011	010	001	000
0	1	0	1	1	0	1	0

binary 01011010 = decimal 90

It is possible to express the evolution of a CA by $S(t+1) = T_R \times S(t)$, where S(t) is the global state of CA at time t and T_R is the transition matrix. For 1-D CA, S is a column matrix of length N and T_R is an NxN matrix. For 2-D CA, S is a column matrix of length N^2 and T_R is an $N^2 \times N^2$ matrix. Below is given an example showing the matrix equation for a 6 cell, rule-90, 1-D CA with null boundaries :

$$\begin{bmatrix} S1(t+1) \\ S2(t+1) \\ S3(t+1) \\ S4(t+1) \\ S5(t+1) \\ S6(t+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S1(t) \\ S2(t) \\ S3(t) \\ S4(t) \\ S5(t) \\ S6(t) \end{bmatrix}$$

In a similar way neighbourhoods and types of behaviour can be expressed for 2-D CA, though these have received less attention in the literature. Examples of the von Neumann and Moore neighbourhoods are shown in Figure 4. Names for such neighbourhoods are according to (Tsalides et al).

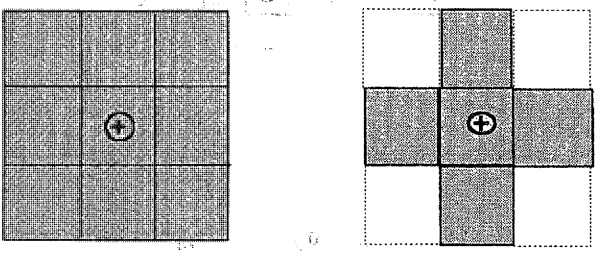

Fig. 4. 2-D neighbourhoods.

3. Mod-p Multiplication Based on 2-D CA

Every element of a cyclic group is a 'power' of one element 'g' - called a generator.

$$G = \{g^0, g^1, g^2, ..., (g^k = I)\}$$

It is well known that the integers form a cyclic group of order 'p-1' (where 'p' is prime) under mod-p multiplication. For instance;

$$g = 3, p = 7 : G = \{1, 3, 2, 6, 4, 5, 1, ...\}$$

This produces the following multiplication table.

Table 1. Mod-7 multiplication using generator '3'.

x	1	3	2	6	4	5
1	1	3	2	6	4	5
3	3	2	6	4	5	1
2	2	6	4	5	1	3
6	6	4	5	1	3	2
4	4	5	1	3	2	6
5	5	1	3	2	6	4

It is also well known that cyclic groups of equal order are isomorphic, in other words there is a one-to-one mapping between elements of the groups. Therefore, in order to multiply two binary numbers (a x b) mod-p, where p is prime, using CA:

i) Use two CA, (A and B) each displaying cyclic groups of order n = p-1

ii) Encode a,b into f(a) and f(b)

iii) Load A with f(a)

iv) Load B with f(1)

v) Cycle both CA until the global state of B = f(b)

vi) The result is then given by the global state of A

The example below shows the multiplication 6 x 4 (mod-7) using rule 150 CA of length 4 and group order n = p-1 = 6, as shown in Figure 1.

ISOMORPHIC MAPPING $\Rightarrow$

$1 = 0\ 1\ 0\ 0$
$3 = 1\ 1\ 1\ 0$
$2 = 0\ 1\ 0\ 1$
$6 = 1\ 1\ 0\ 1$
$4 = 0\ 0\ 0\ 1$
$5 = 0\ 0\ 1\ 1$

Wyższa Szkoła Inżynierska
Instytut Robotyki
i Inżynierii Oprogramowania
ul. Podgórna 50
65-246 Zielona Góra

ITERATIONS ⇒	B				A			
	0	1	0	0	1	1	0	1
	1	1	1	0	0	0	0	1
	0	1	0	1	0	0	1	1
	1	1	0	1	0	1	0	0
	0	0	0	1	1	1	1	0
	0	0	1	1	0	1	0	1

The one-to-one mapping between the states of the CA which form a cyclic group of order 6 and the elements of the mod-7 multiplication using generator '3' is shown above. It can be seen that, for this example, the identity state is assumed to be '0100' and, prior to each calculation, this is loaded into B. At the same time A is loaded with the state corresponding to number '6' (1101). After 4 iterations B contains the state which corresponds to number '4' (0001). The result can be read from A (1110) and decoded into number '3', i.e. 6x4 (mod 7) = 3.

4. Implementation of a Mod-127 Multiplier

For reasons of economy it was decided to implement a mod-127 multiplier using this approach and while, admittedly, it is of modest proportions, such an approach can reveal the potential of the technique for more ambitious devices. (Tsalides et al) have shown that the lower bound for the array size 'N' to yield group order 'n' for 2-D CA which evolve according to rule (4,2-D) using the von Neumann neighbourhood using such CA is given by :

$$n = 2 (2^{N/2} - 1)$$

Therefore, for group order n=126, the multiplier is based on two 12x12 CA with null boundaries. It is quite evident that such a scheme has huge redundancy as only 126 states are used from a total of 2^{144}. For this reason the data compression capabilities of CA, which arise due to the occurence of symmetrical states, have been exploited.

If the initial state of the present CA is symmetrical then it is clear that all subsequent states in the cycle will also be symmetrical due to the nature of the local rule. In the present case 8-fold symmetry has been employed and this enables the use of 15-cell 'triangular CA' as shown in Figure 5.

Because of symmetry the cells that lie on the diagonal of the array maintain their initial state during the evolution of the array. Each remaining cell is associated with 7 'twin' cells in the other octants, depicted by 'a, b, c, ...n, o' that display the same local state. Therefore reduced 'triangular' CA, consisting of 15 cells can be employed with boundary conditions as indicated in Figure 5.

Null boundaries are maintained along the original edge of the CA. A 'mirror' boundary is formed along the edge containing cells 'e, i, l, n, o' because each cell in such a position communicates with its 'twin' from the neighbouring octant. The diagonal boundary has no effect because the two 'I' cells neighbouring 'a, f, j, m, o' are

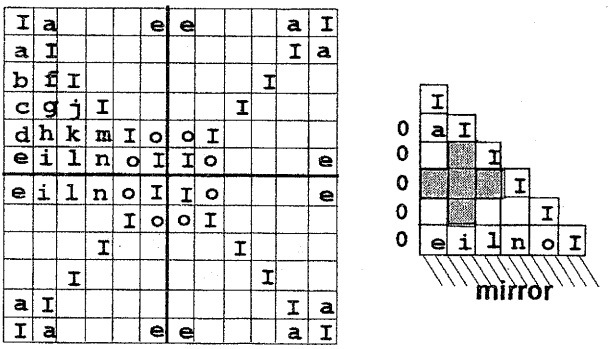

Fig. 5. Triangular CA.

arranged to have identical contents. The group order of such triangular arrays is halved, to 63, due to symmetry and therefore an overflow bit is also employed in order to realise 126 distinct states. This results in considerably reduced, though admittedly still considerable, redundancy, 126 states are employed from a possible 2^{16}.

In order to emulate a system that can communicate with the outside world it is necessary to use binary communications. This could be achieved by the use of a 7-bit to 16 bit encoder for the inputs and a 16-bit to 7 bit decoder for the outputs. In order to reduce the demands on silicon an investigation was undertaken to seek optimum 6-bit signatures, each of which unambiguously represents a global state of the CA. From 4000 computer simulations of the triangular CA, each starting from a unique initial state, only 4 yielded such a signature. This has enabled the use of a reduced decoder having 7-bit input (6-bit signature + overflow) and 7-bit output. In addition, from observation it was found that the binary representations of the first 63 states are complementary to those for the second 63 states. Therefore, with careful selection of inverted outputs, only half as many product terms are required for the decoder. Both the encoder and decoder have been realised as Programmable Logic Arrays (PLA).

4.1 Full Custom Implementation

The mod-127 multiplier has been designed using 2 micron CMOS design rules from European Silicon Structures (ES2). CASS and Princess from Silvar Lisco, HSPICE from Meta software and HILO from Genrad were used for schematic capture, layout, circuit simulation and logic simulation respectively. The encoder and decoder have been implemented using pseudo-nMOS Programmable Logic Arrays which offer small size and simplicity but have relatively high power dissipation.

Simulation results for the whole design have been obtained using HILO with gate delays extracted from circuit simulation of the cells using HSPICE. From these results it was predicted that the multiplier would operate at a maximum clock frequency of 25 MHz. The completed chip occupies an area measuring 2.4 mm x 2.2 mm (including pads) and contains approximately 4,600 transistors. It was fabricated, by arrangement

with the UK ECAD initiative, at a cost of £100 per mm^2 with a turnaround time of 12 weeks. Tests, using Hewlett-Packard equipment and a breadboard, reveal 100% yield up to a clock frequency of 20 MHz. It should be noted that the nature of the architecture leads to a worst case calculation time of 126 clock cycles.

A floorplan for the multiplier is shown in Figure 6. A schematic showing the repeated CA cell is shown in Figure 7 and the completed layout is depicted in Figure 8.

4.1.1 VLSI Complexity

Using the VLSI compexity metrics, A, AT and AT^2, of Thompson (1979) previous authors (Pries, Thanailakis and, 1986) have compared the CA approach with more conventional algorithms. Results are presented in Figure 9. It can be seen from this figure that the 2-D CA with encoder and decoder does not offer the best solution, using these metrics, as the problem size grows. However, it should be remembered that the modularity, regularity and local interconnection strategy offer advantages in terms of design complexity and this may be a significant factor.

4.2 Programmable Gate Array Implementation

The multiplier has also been implemented using a Xilinx LCA 2018 programmable gate array. This device contains approximately 1800 'gate equivalents' and costs about £10. Design was performed on a PC-compatible using Orcad for schematic capture and simulation and XACT for automatic layout. The chip is configured at power-up using an EPROM.

In order to reduce the number of required gates this implementation employs a slightly different strategy to that used for the full custom approach. At the initiation of a calculation binary input 'a' is first latched into the chip and then CA 'A' is cycled until its decoded state corresponds to 'a'. This removes the need for the area hungry encoder, at a cost of decreased performance. The completed circuit operates at a clock frequency of 10 MHz and this is limited by the performance of the chip. Xilinx arrays include a number of active switches in each interconnection path and these impose significant delays on signals. For this reason Xilinx LCAs typically operate at system clock frequencies that are only about 20% of the specified performance, which in the case of the present design is 50 MHz.

5. Summary

The work has demonstrated the suitability of CA for silicon implementations. Although the design is of modest proportions the full custom implementation can be regarded as 'VLSI' in the sense that it uses appropriate CAD tools and techniques. VLSI complexity measures relating area and performance of a mod-p multiplier to problem complexity are dissapointing but advantages in terms of design complexity are evident. The implementation using 2-D CA incurs significant redundancy, despite the use of 'triangular' CA, arising from considerations of symmetry, to reduce the necessary array size by about 90%. More recent work has concentrated on 'hybrid' CA in which the

local rule is not unique across the array and this has yielded optimum sized 1-D arrays with group order 'p-1' and no redundancy. This latter work has been submitted for publication elsewhere and is currently under consideration.

Acknowledgements

The authors are grateful for the support of NATO Research grant RG 0835/88 and for the facilities made available by the UK ECAD initiative.

References

Fraenkel A.S. (1961): *Use of index calculus and mersenne primes for the design of a high-speed digital multiplier.*- J. Ass. Comp. Mech., v.8, No.1, p.87.

Huberman B.A. and Hogg T. (1984): *Adaptation and self-repair in parallel computing structures.*- Phys. Rev. Lett., v.59, No.12, p.1048.

Jenkins W.J. and Leon F.J. (1977): *The use of residue number system in the design of finite impulse response filters.*- IEEE Trans. Circuits and Systems, v.24, No.4, p.191.

Margulus N., Toffoli T. and Vichiac G. (1986): *Cellular automata supercomputers for fluid-dynamics modelling.*- Phys. Rev. Lett., v.56, No.16, p.1694.

Neumann J.Von (1966): *Theory of Self-Reproducing Automata.*- University of Illinois Press, Urbana.

Pries W., Thanailakis A. and Card H.C. (1986): *Group properties of cellular automata and VLSI applications.*- IEEE Trans. Comp., v.35, No.12, p.1013.

Soderstrand M.A. (1977): *A high-speed low-cost recursive digital filter using residue number arithmetic.*- Proc. IEEE, v.65, p.1065.

Soderstrand M.A. (1983): *A new hardware implementation of modulo adders for residue number systems.*- Proc. of the 26th Midwest Symposium on Circuits and Systems, p.412.

Soderstrand M.A., Jenkins W.K., Jullien G.A. and Taylor F.J. (1986): *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing.*- IEEE Press.

Soderstrand M.A. and Vernia C. (1980): *A high-speed low-cost modulo P_i multiplier with RNS arithmetic applications.*- Proc. IEEE, v.68, p.529.

Taylor F.J. (1982): *A VLSI residue arithmetic multiplier.*- IEEE Trans. Comp., v.31, No.6, p.540.

Taylor F.J. and Huang C.H. (1982): *An autoscale residue multiplier.*- IEEE Tran. Comp., v.31, No.4, p.321.

Thompson C.D. (1979): *Area-time complexity for VLSI.*- Proc. 11th Annual ACM Symposium on the Theory of Computing, p.81.

Tsalides Ph., Pitsianis N., Bleris G.L. and Thanailakis A.: *Two-dimensional cellular automata: properties and applications of a new VLSI architecture.*- Accepted for publication in Computer J. of the British Computer Society.

Tsalides Ph., Thanailakis A., Card H.C. and Pries W. (1987): *Two-dimensional cellular automata as VLSI function blocks and their computer applications.*- IEEE Comp. Euro, p.160.

Tseng B.D., Jullien G.A. and Miller W.C. (1979): *Implementation of FFT structures using the residue number system.*- IEEE Trans. Comp., v.28, No.11, p.831.

Wyższa Szkoła Inżynierska
Instytut Robotyki
i Inżynierii Oprogramowania
ul. Podgórna 50
65-246 Zielona Góra

Wolfram S. (1986): *Theory and Applications of Cellular Automata.*- World Scientific Publishing Co. Pte. Ltd., Singapore.

Wolfram S. (1983): *Statistical mechanics of cellular automata.*- Rev. Mod. Phys., v.55, No.3, p.601-644.

Yau S.S. and Chung J. (1976): *On the design of modulo arithmetic units based on cyclic groups.*- IEEE Trans. Comp., v.25, No.11, p.1057.