Robust functional testing for VLSI cellular neural network implementations

Citation for published version (APA):

Grimaila, M. R., Pineda de Gyvez, J., & Han, G. (1997). Robust functional testing for VLSI cellular neural network implementations. *IEEE Transactions on Circuits and Systems. I, Fundamental Theory and Applications*, 44(2), 161-166. <https://doi.org/10.1109/81.554336>

DOI:

[10.1109/81.554336](https://doi.org/10.1109/81.554336)

Document status and date:

Published: 01/01/1997

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Fig. 5 illustrates that doubling the number of segments per unit distance to 1000/50 cm (while using a 150 V triangular pulse at the input) gives less attenuation, in support of the theory that as a transmission line is approached, solitons propagate without excessive attenuation. Fig. 6 shows that if the input amplitude goes to 300 V, while keeping the 100 ps transitions, two solitons result assuming 1000/50 cm, as noted in [9]. This shows that it is important to employ appropriate inputs, and an ample number of segments per unit distance, more than suggested for simple edge sharpening in [4] or [6].

V. CONCLUDING COMMENTS

Simulations can have nonlinear inductance and can include frequency-dependent parameters using the above method. Computer advances in conjunction with a simple algorithm render the method quite accessible. Furthermore, it is possible to extend the method to nonuniform lines involving abrupt bends or encounters with other objects.

APPENDIX A

Differentiation equation

$$s^n F(s) = \int_{0-}^{\infty} \frac{d^n}{dt^n} f(t) e^{-st} dt \quad (A1)$$

where n is a positive integer, and $F(s)$ is a Laplace transform of $f(t)$. For example, $\sqrt{s}$ can be expressed as $s/\sqrt{s}$ where from Table I in the text (using $\lambda = -0.5$) the inverse Laplace transform of $1/s^{0.5}$ is $1/\sqrt{\pi t}$; (A1) readily shows that the transform of $\sqrt{s}$ is $1/(2\sqrt{\pi t}^{1.5})$ which agrees with Table I using $\lambda = -1.5$.

REFERENCES

- [1] N. Seddon and E. Thornton, "A high-voltage, short risetime, pulse sharpener," *Rev. Sci. Instrum.*, vol. 59, no. 11, pp. 2497–2498, Nov. 1988.
- [2] C. R. Wilson, M. M. Turner, and P. W. Smith, "Pulse sharpening in a uniform LC ladder network containing nonlinear ferroelectric capacitors," *IEEE Trans. Electron Devices*, vol. 38, pp. 767–771, Apr. 1991.
- [3] T. Kuusela and J. Hietarinen, "Nonlinear electrical transmission line as a burst generator," *Rev. Sci. Instrum.*, vol. 62, pp. 2266–2270, Sept. 1991.
- [4] M. M. Turner, G. Branch, and P. W. Smith, "Methods of theoretical analysis and computer modeling of the shaping of electrical pulses by nonlinear transmission lines and lumped-element delay lines," *IEEE Trans. Electron Devices*, vol. 38, pp. 810–815, Apr. 1991.
- [5] R. J. Baker *et al.*, "Design note—Generation of kilovolt-subnanosecond pulses using a nonlinear transmission line," *Meas. Sci. Technol.*, vol. 8, pp. 893–895, Aug. 1993.
- [6] J. R. Burger, "Modeling solitary waves on nonlinear transmission lines," *Trans. Circuits Syst. I*, vol. 42, pp. 34–36, Jan. 1995.
- [7] R. Hirota and K. Suzuki, "Theoretical and experimental studies of lattice solitons in nonlinear lumped networks," *Proc. IEEE*, vol. 61, pp. 1443–1491, Oct. 1973.
- [8] T. Kuusela, "Soliton experiments in a damped ac-driven nonlinear electrical transmission line," *Phys. Lett. A*, vol. 167, no. 1, pp. 54–59, July 1992.
- [9] T. Tsuboi, "Formation process of solitons in a nonlinear transmission line: Experimental study," *Phys. Rev. A*, vol. 41, no. 8, pp. 4534–4537, Apr. 1990.
- [10] M. C. Gutzwiller and W. L. Miranker, "Nonlinear wave propagation in a transmission line loaded with thin permalloy films," *IBM J. Res. Develop.*, vol. 7, pp. 278–287, Oct. 1963.
- [11] H. A. Haus, "Molding light into solitons," *IEEE Spectrum*, pp. 48–53, Mar. 1993.
- [12] G. Liu *et al.*, "A method for calculating responses of sub-nanosecond circuits with nonlinear components," in *Proc. 34th Midwest Symp. Circuits Syst.*, Monterey, CA, 1991, pp. 186–190.

Robust Functional Testing for VLSI Cellular Neural Network Implementations

Michael Russell Grimaila, Jose Pineda de Gyvez, and Gunhee Han

Abstract—A robust testing method for detecting circuit faults within two-dimensional Cellular Neural Network (CNN) arrays is presented. The functional tests consist of a sequence of input vectors that toggle all internal nodes of the conceptual CNN model and propagate the result to the output pins. The resultant output vectors reveal nodes that exhibit opened, shorted, or stuck-at faults. The generated test vectors are universal, detect faults independent of the size or topology of the CNN array, and can be applied to any particular CNN implementation with little effort.

Index Terms—Cellular Neural Networks, Testing, C-Testability.

I. INTRODUCTION

Cellular Neural Networks (CNN's) are a special class of two-dimensional, continuous-time analog signal processor arrays [1]. Several CNN implementations have emerged as a result of the encouraging flexibility and speed exhibited in distinct applications [2]–[8]. Despite the large number of CNN implementations that have been developed, the problem of testing has not been addressed properly [9]–[12]. The inaccessibility of nodes within VLSI CNN implementations makes testing more difficult, increasing both the time and cost required to functionally verify the integrity of the hardware. In this brief, we present a systematic method to detect functional faults within a CNN array using a sequence of functional tests. Each functional test isolates a unique path through the array and exercises the path to identify nodes that are open, shorted, or stuck-at a supply voltage. The cells within a CNN array are internally identical and vary only in the interconnection to their neighbors. For this reason, techniques based on C-Testability [13] are used whenever possible which allow cells to be tested in parallel and independent of their location within the array. The tests provide the necessary testing regime to insure baseline functionality in VLSI CNN implementations.

II. GENERIC STRUCTURE FOR CELLULAR NEURAL NETWORKS

For the sake of clarity, we are going to present our test methodology based upon a generic CNN abstract model [14]. However, all of the test vectors are applicable to actual VLSI hardware as shown in Sections V and VI. A generic CNN architecture [15] is shown in Fig. 1(a)–(d). Basically, this architecture embeds all the electrical and behavioral properties of actual CNN implementations. Although physical implementations may differ in the actual number and type of mode control signals, the behavioral steps discussed next are common to the proper operation of all CNN's. The CNN array is controlled by choosing the initial conditions, setting the external inputs, and selecting the desired mode of operation. Three behavioral modes of operation can be identified: i) *Cell Initialization*, ii) *Cell Evaluation*, and iii) *Result Extraction*. In our generic model, behavioral modes are selected by two mutually exclusive signals, the *set initial condition* signal, S , and the *evaluate* signal, E . In the initialization phase, S

Manuscript received September 4, 1995; revised April 2, 1996. This work was supported by the Office of Naval Research under Grant N00014-91-1-0516. This paper was recommended by Associate Editor H. P. Graf.

The authors are with the Department of Electrical Engineering, Texas A&M University, College Station, TX 77843 USA.

Publisher Item Identifier S 1057-7122(97)00809-X.

Fig. 1. CNN Cell Functional testing paths. (a) Initial condition. (b) Bias. (c) Input. (d) Feedback.

is activated and E is negated allowing the state of the cell to be charged to the desired initial condition value X_0 . The CNN enters the evaluation mode when E is activated and S is negated, connecting the summer to the integrator and allowing the dynamic processing to commence. After the cell converges, or after a finite period of time, both S and E signals are negated which disconnects the summer from the integrator, holding the cell state for the result extraction phase. We can write the CNN's state dynamics in a descriptive form that includes the effect of the control signals as follows:

$$\begin{aligned} \{x_{ij}\}^{\langle \bar{S}, \bar{E} \rangle} &= \{x_{ij}(0)e^{-t/RC}\}^{\langle \bar{S}, \bar{E} \rangle} \\ &+ \left\{ \frac{1}{C} \int_0^t e^{-(t-\phi)/RC} \left[\sum_{C(k,l) \in N(i,j)} A(i,j;k,l)y_{kl}(t) \right. \right. \\ &\left. \left. + \sum_{C(k,l) \in N(i,j)} B(i,j;k,l)u_{kl} + I \right] d\phi \right\}^{\langle \bar{S}, \bar{E} \rangle} \quad (1) \end{aligned}$$

where the superscripts $\langle \cdot, \cdot \rangle$ are used to indicate the controlling conditions under which the distinct sections of the equation are executed in the actual hardware. The equation clearly shows that the state of the cell depends upon the mode of operation.

III. FUNCTIONAL FAULT TESTING STRATEGY

The proposed testing methodology requires visibility of the post-activation function output y_{ij} , and optionally the state x_{ij} . The input vectors were designed so that every node in the conceptual CNN model is toggled and the result is propagated to these observable nodes. Although the tests cannot isolate the type or exact location of

the fault with only these two observable nodes, they do identify the path that contains the fault(s). The functional tests are designed to remove any dependence upon the cell time constant τ , thus the test vectors are applied at intervals much greater than the time constant of the CNN cell.

In an effort to simplify the complexity of the testing problem, four different hardware testing paths in the CNN architecture were identified. The effect of this observation is that the functional tests are divided into four segments, one for each of the testing paths as shown in Fig. 1(a)–(d). The corresponding testing equations are the initial condition path (2a), the biasing path (2b), the input path (2c), and the feedback path (2d). Their mathematical origin comes from splitting (1) as follows

$$\text{Initial condition path} = \{x_{ij}(0)e^{-t/RC}\}^{\langle \bar{S}, \bar{E} \rangle} \quad (2a)$$

$$\text{Bias path} = \left\{ \frac{1}{C} \int_0^t e^{-(t-\phi)/RC} (I) d\phi \right\}^{\langle \bar{S}, \bar{E} \rangle} \quad (2b)$$

$$\begin{aligned} \text{Input path} &= \left\{ \frac{1}{C} \int_0^t e^{-(t-\phi)/RC} \right. \\ &\quad \times \left[\sum_{C(k,l) \in N(i,j)} B(i,j;k,l)u_{kl} \right] d\phi \left. \right\}^{\langle \bar{S}, \bar{E} \rangle} \quad (2c) \end{aligned}$$

$$\begin{aligned} \text{Feedback path} &= \left\{ \frac{1}{C} \int_0^t e^{-(t-\phi)/RC} \right. \\ &\quad \times \left[\sum_{C(k,l) \in N(i,j)} A(i,j;k,l)y_{kl}(t) \right] d\phi \left. \right\}^{\langle \bar{S}, \bar{E} \rangle} \quad (2d) \end{aligned}$$

The lack of summations in (2a)–(2b) imply that the initial condition and biasing paths of each cell can be tested in parallel with no interaction between adjacent cells. During these tests, every element in the A and B templates is set to zero eliminating any interaction between adjacent cells. The presence of the summation operators in (2c)–(2d) requires that only one nonzero template value be chosen when testing the input and feedback paths. Each test vector uses only one nonzero template entry to isolate a single path through a multiplier and the cell interconnection between adjacent cells.

IV. FUNCTIONAL FAULT TESTING PRELIMINARIES

In an effort to clarify the testing strategy, we first define some terminology. In our analysis, we consider three static fault models: i) *Short*—A low impedance fault less than 1.0Ω is connected between two nodes ii) *Open*—A node is broken into two electrically isolated subnodes iii) *Stuck At*—A node is shorted to V_{DD} , GND , or V_{SS} . All inputs applied to the array during the functional tests are contained in the normalized set $\{-1, 0, 1\}$ which correspond to the colors white, gray, black. The input values must be scaled when considering the application of the testing sequence to each particular CNN implementation. The tests must be applied in the sequence specified in this section. If a fault is detected, the testing sequence is terminated and the array is declared faulted.

A. Vector Generation for Initial Condition Path

The purpose of this test is to determine if the initial condition can be properly initialized and extracted. The test verifies that each cell's switch, SW_2 , state node, x_{ij} , and the activation functional output, y_{ij} , are fault free. The test requires manipulation of the control signal S and of the initial condition signal X_o . The feedback template, the control template, the I value, and the external input values u_{ij} are all set to 0 and the control signal E is negated to prevent the cell from being influenced by the multiplier or bias circuits. The initial condition is set to the desired value by applying the desired dc value to the X_o input and by activating S . The effect is that the X_o value charges the state capacitor to the desired value $x(0)$. Once the cell's state is stable, the array is then switched to the result extraction mode by deactivating S . Ideally the cell state would be held indefinitely, but in real circuit implementations the cell state drifts due to offsets and leakage. The length of time that a cell can hold a result, within a certain percent deviation, is used as a figure of merit. In cell addressable architectures without parallel output capability, this will determine the maximum time between starting and completing the result extraction phase. Typically, the designer will specify a maximum discharge rate used to determine if a cell is "bad". The following pseudocode summarizes this strategy

```

 $A^{(\bar{S}, \bar{E})} \leftarrow 0, B^{(\bar{S}, \bar{E})} \leftarrow 0, I_{Bias}^{(\bar{S}, \bar{E})} \leftarrow 0$ 
 $u_{ij}^{(\bar{S}, \bar{E})} \leftarrow 0$ 
 $[x_{ij}(0)]^{(\bar{S}, \bar{E})} \leftarrow 0$ 
do  $\{ [x_{ij}(0)]^{(\bar{S}, \bar{E})} \leftarrow \{-1, 0, 1\} \}$ 
     $[x_{ij}(t)]^{(\bar{S}, E)}$  run CNN
     $\forall ([x_{ij}(t_\infty)]^{(\bar{S}, \bar{E})} \neq x_{ij}(0)) C_{ij} \Rightarrow \text{faulty}$ 
end
```

B. Vector Generation for Bias Path

The purpose of this test is to determine if the bias path contains any faults. The test verifies that the I input, the summer, switch SW_1 , and the summer to the integrator path do not contain any static faults by ensuring that the I signal correctly influences the output of each

cell. The test requires manipulation of the control signals S and E as well as the bias input, I . The A and B templates and the external inputs u_{ij} are set to 0. All cells are initialized to 0 and the array is placed in evaluate mode by asserting the E signal. The I input signal is changed from 1, to 0, to -1 . When the I input signal is 0, the offset for each cell in the array is measured and recorded. The collected data can be used later to null the offset in the system. The output of the cell is monitored to insure it follows the I input. The pseudocode for generating the test vectors is shown below.

```

 $A^{(\bar{S}, \bar{E})} \leftarrow 0, B^{(\bar{S}, \bar{E})} \leftarrow 0$ 
 $u_{ij}^{(\bar{S}, \bar{E})} \leftarrow 0$ 
 $[x_{ij}(0)]^{(\bar{S}, \bar{E})} \leftarrow 0$ 
do  $\{ [I_{Bias}]^{(\bar{S}, E)} \leftarrow \{-1, 0, 1\} \}$ 
     $[x_{ij}(t)]^{(\bar{S}, E)}$  run CNN
    if  $(I_{Bias} == 0)$  record cell offset
         $\forall ([x_{ij}(t_\infty)]^{(\bar{S}, \bar{E})} \neq 0) C_{ij} \Rightarrow \text{offset}$ 
         $\forall ([x_{ij}(t_\infty)]^{(\bar{S}, \bar{E})} \neq [I_{Bias}]) C_{ij} \Rightarrow \text{faulty}$ 
end
```

C. Vector Generation of Input Path

The purpose of this test is to verify that the input image path, the control multipliers, and the multiplier to summer paths do not contain any faults. For this test, template A and the I value are set to 0 and the E signal is activated. The input path is tested by applying a fixed input pattern, u , and setting one nonzero entry of the B template to 1. These tests directionally propagate the input image through each possible cell interconnection in the input path. The output of the array is sampled and compared to the expected output image. This first step of the test verifies only one quadrant of the multiplier's operation. Next, the nonzero B template entry is switched from 1 to -1 to verify the second quadrant of the selected input path multiplier. Finally, the input image is changed to its complement and the above test is repeated to verify the final two quadrants of the four quadrant multiplier. The expected output, y_{ij} , depends on the B template, the input image, and the location of the cell within the array. For full test coverage, the same procedure is applied for each of the nine entries of the B template. The pseudocode for this test segment is shown below.

```

 $\aleph$  set of input images
 $\Im$  set of output images
 $A^{(\bar{S}, \bar{E})} \leftarrow 0, I_{Bias}^{(\bar{S}, \bar{E})} \leftarrow 0$ 
 $[x_{ij}(0)]^{(\bar{S}, \bar{E})} \leftarrow 0$ 
do  $\{ [B(i, j; k, l)]^{(\bar{S}, \bar{E})} \leftarrow \{1, -1\} \quad \forall (k = -1, 0, 1 \quad l = -1, 0, 1) \}$ 
     $u_{ij}^{(\bar{S}, \bar{E})} \leftarrow \aleph$ 
     $[x_{ij}(t)]^{(\bar{S}, E)}$  run CNN
     $\forall ([y_{ij}(t_\infty)]^{(\bar{S}, \bar{E})} \notin \Im) C_{ij} \Rightarrow \text{faulty}$ 
end
```

D. Vector Generation of Feedback Path

The purpose of this test is to determine if the feedback path, the feedback multipliers, and the multiplier to summer paths contain any faults. Feedback path testing is more difficult to conduct because there is not independent control of the inputs to each multiplier. Nevertheless, it is possible to carry out these tests in an effective manner by dividing the tests into two segments: neighbor feedback and self feedback. The neighbor feedback tests are similar to the input path testing described in the previous section. The feedback

Fig. 2. VLSI CNN Cell System Level Schematic with BVCC Expanded.

Fig. 3. Bias Path Test Output Vectors. (a) The unfaulted state voltage response in cell (3, 3). (b) The state voltage response when fault #2 is introduced into the Bias Voltage to Current Converter by shorting the gate of M3 and M4 to VSS in cell (3, 3).

Fig. 4. Input Path Test Output Vectors. (a) The unfaulted state voltage response in cell (4, 4). (b) the state voltage response in cell (4, 4) when fault #2 is introduced by opening the output of multiplier B00 in cell (3, 3).

path involving the template value A_{11} must be tested independently due to the self-feedback induced.

For the neighbor feedback tests the initial condition value, $x(0)$, and the I values are set to 0. The E signal is held in the active state to propagate the input image and is negated when extracting results. To avoid the need of external inputs for the periphery cells, we use the B template to apply the input image to cells located on the boundary of the array. Thus, the input path test segment must be completed successfully before this test may be considered valid. The external inputs to cells on the boundary of the array provide stable values which are propagated to the adjacent cells via the feedback multipliers. The stable values can then be used as the nonaccessible internal input of the A multipliers, eliminating the need for a dedicated border value. The feedback path is tested by applying a fixed input pattern, u_{ij} , setting the center element of the B template to 1, and setting one nonzero entry of the A template to 1. The input image has all entries fixed to zero except those acting as sources for the A template directional propagation. The expected output image is the result of the applied input propagated across the array. Next, the nonzero A template entry is switched from 1 to -1, and the input image is inverted to verify the four quadrants of operation of the selected feedback path multiplier. The expected output, y_{ij} , depends on the A template, B template, the input image, and the location of the cell within the array. As with the B template, full test coverage requires exercising each entry of the A template using the previous procedure.

Testing the self feedback path is based on the premise that when an initial condition is set and a positive feedback term slightly greater than 1 is chosen, the state of the cell will saturate to the supply rail corresponding to the sign of the initial condition value. Conversely, when a negative feedback term is chosen, the state of the cell will converge to zero because $A_{ii} < 1/R$. When testing the self-feedback

path, a nonzero initial condition value, $x(0)$, is present along with the desired A_{11} element. The B template and the $Bias$ values are set to 0. The E signal is held in the active state until the cell converges. The expected cell output is a saturated state for positive feedback values and zero for negative feedback values. The corresponding pseudocode is shown below.

```

 $\aleph$  set of input images
 $\Im$  set of output images
 $I_{Bias}^{(\bar{S}, \bar{E})} \leftarrow 0$ 
do {  $[A(i, j; k, l)]^{(\bar{S}, \bar{E})} \leftarrow \{1, -1\} \forall (k = -1, 0, 1 \ l = -1, 0, 1)$  }
    if  $(k \neq 0 \wedge l \neq 0)$  { neighbor feedbacks
         $B(i, j; k, l)^{(\bar{S}, \bar{E})} \leftarrow 1$ 
         $u_{ij}^{(\bar{S}, \bar{E})} \leftarrow \aleph$ 
         $[x_{ij}(0)]^{(\bar{S}, \bar{E})} \leftarrow 0$ 
    }
    else { self feedback
         $B(i, j; k, l)^{(\bar{S}, \bar{E})} \leftarrow 0$ 
         $u_{ij}^{(\bar{S}, \bar{E})} \leftarrow 0$ 
         $[x_{ij}(0)]^{(\bar{S}, \bar{E})} \leftarrow \{-1, 1\}$ 
    }
     $[x_{ij}(t)]^{(\bar{S}, \bar{E})}$  run CNN
    if  $(k \neq 0 \wedge l \neq 0)$  neighbor feedbacks
         $\forall ([y_{ij}(t_\infty)]^{(\bar{S}, \bar{E})} \notin \Im) C_{ij} \Rightarrow \text{faulty}$ 
    else self feedback
         $\forall ([y_{ij}(t_\infty)]^{(\bar{S}, \bar{E})} \notin \{0, 1\}) C_{ij} \Rightarrow \text{faulty}$ 
end

```

V. A VLSI CELLULAR NEURAL NETWORK IMPLEMENTATION

In this section, we briefly describe an actual VLSI CNN implementation fabricated in MOSIS 2- μm CMOS [16]. The VLSI implementation is used as the actual test vehicle for our investigation. Each cell in the VLSI CNN array contains 18 multipliers, a lossy integrator, a hard limiter, a biasing circuit, and a sample/hold circuit as shown in Fig. 2. All of the multipliers are located at the output of the cell, as opposed to the generic CNN, where the multipliers are located at the input. The power supplies are ± 3 V and the state voltage is bounded by these values. The hard limiter implements the activation function which saturates at ± 0.5 V. The array output is the state voltage and requires an external activation function to limit in the range $\{-0.5, 0.5\}$. The integration capacitance is 1 pF and the OTA resistor is adjustable in the range from 141 k Ω to 2 M Ω . The transconductance factor for the multipliers, gm_{MULT} , is 20 $\mu\text{A/V}$ and the transconductance factor for the bias circuit, gm_{BIAS} , is 8.5 $\mu\text{A/V}$.

VI. TESTING A REAL VLSI CNN IMPLEMENTATION

In this section, two mutually exclusive and randomly chosen static faults are introduced into the transistor model of a 5×5 VLSI CNN array described in Section V. The testing methods presented in this brief are applied to the faulted array and simulated using HSPICE. After introducing each fault, the array is simulated and the corresponding path is examined to verify the fault detection capability of the proposed method. All of the faults considered are introduced into the center cell of the array, i.e., $C(3,3)$. In the following discussion, refer to Fig. 2 for the locations of the faults described.

Fault #1 is introduced into the bias path by shorting the gate of M3 and M4 to V_{SS} in the Bias Voltage to Current Converter (BVCC) creating a stuck-at fault. Fig. 3 shows the unfaulted and faulted output curves for the bias path test. In Fig. 3(a), the state voltage correctly follows the bias input. The positive and negative peak voltages induced at time 1.5 μs and 2 μs meet the required 0.5 V state voltage deflection. In Fig. 3(b), the state voltage of the cell does not follow the bias voltage and instead drifts negative due to the uncompensated cell offset. The state voltage remains fixed at -0.247 volts and is unaffected by the input because of the fault introduced into the BVCC.

Fault #2 is introduced in the input path by opening the output of multiplier B00. A checkerboard input image is applied and the B00 template is fixed to 0.5 V. Since the fault was introduced in cell (3, 3), we must examine the output from cell (4, 4) due to the selected direction of propagation. Fig. 4 shows the unfaulted and faulted output curves. The correct output image is the input image when propagated down and to the right as shown in Fig. 4(a). Gray cells indicate intermediate values between black and white. Since cells on the perimeter of the array do not have inputs from outside the boundary of the array, they converge to a gray value. Fig. 4(b) shows the faulted state voltage from cell (4, 4) resulting from opening multiplier B00's output in cell (3, 3). Ideally the faulted state voltage would be zero, but the accumulation of offsets from each of the multipliers in surrounding cells contributes to the nonzero state value.

VII. CONCLUSION

In this brief, we have presented a robust strategy for functional testing hardware implementations of CNN's. The testability of the CNN was clarified by dividing the cell into four different testing paths. We have shown that the tests provide extensive fault coverage independent of the array size or topology without the need for additional hardware.

REFERENCES

- [1] L. Chua and L. Yang, "Cellular neural networks: Theory," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1257–1272, 1988.
- [2] —, "Cellular neural networks: Applications," *IEEE Trans. Circuits Syst.*, vol. 35, pp. 1273–1290, 1988.
- [3] K. R. Krieg, L. O. Chua, and L. Yang, "Analog signal processing using cellular neural networks," in *Proc. IEEE ISCAS*, 1990, pp. 958–961.
- [4] J. M. Cruz and L. Chua, "Design of high-speed, high-density CNN's in CMOS technology," *Int. J. Circuit Theory Applicat.*, vol. 20, pp. 555–572, 1992.
- [5] J. E. Varrientos, E. Sanchez-Sinencio, and J. Ramirez-Angulo, "A current-mode cellular neural network implementation," *IEEE Trans. Circuits Syst. II*, vol. 40, pp. 147–155, 1993.
- [6] T. Roska and L. O. Chua, "The CNN universal machine: An analogic array computer," *IEEE Trans. Circuits Syst. II*, vol. 40, pp. 163–173, 1993.
- [7] I. A. Baktir and M. A. Tan, "Analog CMOS implementation of cellular neural networks," *IEEE Trans. Circuits Syst. II*, vol. 40, pp. 200–205, 1993.
- [8] J. A. Nossek and G. Seiler, "Cellular neural networks: Theory and circuit design," *Int. J. Circuit Theory Applicat.*, vol. 20, pp. 533–553, 1992.
- [9] S. M. Gowda, B. J. Sheu, and J. Choi, "Testing of programmable analog neural network processors," in *Proc. IEEE Custom Integrated Circuits Conf.*, 1992, pp. 17.1.1–17.1.4.
- [10] C. Wey, "Built-In self-test structure for analog circuit fault diagnosis," *IEEE Trans. Instrum. Meas.*, vol. 39, pp. 517–521, June 1990.
- [11] J. Willis and J. Pineda de Gyvez, "Behavioral testing of cellular neural networks," in *ISCAS '94*, vol. 6, Dec. 1994, pp. 229–232.
- [12] A. Rueda and J. L. Huertas, "Testing in analogue cellular neural networks" in *Int. J. Circuit Theory Applicat.*, vol. 20, pp. 583–587, 1992.
- [13] W. K. Huang and F. Lombardi, "C-testability of two dimensional sequential arrays," *Int. J. Electron.*, vol. 69, no. 5, pp. 681–689, 1990.
- [14] M. R. Grimaila, "Comprehensive functional testing and dynamic compensation techniques for cellular neural networks," Master's degree Thesis, Texas A&M Univ., Science Elect. Eng., Dec. 1995.
- [15] M. R. Grimaila and J. Pineda de Gyvez, "A macromodel fault generator for cellular neural networks," in *Proc. IEEE Int. Wkshp. Cellular Neural Networks Applicat.*, CNNA-94, Rome, Italy, 1994, pp. 369–374.
- [16] J. Pineda de Gyvez, "Cellular neural network VLSI implementation," TEES Tech. Rep. vol. 5, no. 93–549/5, May 17, 1995.