# Emulated Digital CNN-UM Implementation of a Barotropic Ocean Model

Zoltán Nagy, Péter Szolgay[+]
Department of Image Processing and Neurocomputing
University of Veszprém
Egyetem u. 10, 8200 Veszprém, Hungary
E-mail: nagyz@almos.vein.hu, szolgay@sztaki.hu

*Abstract*— **The solution of partial differential equations (PDE) has long been one of the most important fields of mathematics, due to the frequent occurrence of spatio-temporal dynamics in many branches of physics, engineering and other sciences. One of the most exciting area is the simulation of compressible and incompressible fluids which appears in many important applications in aerodynamics, meteorology and oceanography. On the other hand the solution of these equations requires enormous computing power. In this paper a CNN-UM simulation of ocean currents will be presented. Unfortunately the non-linearity of the governing equations does not make possible to utilize the huge computing power of the analog CNN-UM chips. To improve the performance of our solution an emulated digital CNN-UM is used where the cell model of the architecture is modified to handle the non-linearity of the model.**

## I. Introduction

A Cellular Neural Network is a non-linear dynamic processor array. Its extended version, the CNN Universal Machine (CNN-UM), was invented in 1993 [1]. The CNN paradigm is a natural framework to describe the behavior of locally interconnected dynamical systems which have an array structure. So, it is quite straightforward to use CNN to compute the solution of partial differential equations (PDE). Several studies proved the effectiveness of the CNN-UM solution of different PDEs [2], [3]. But the results cannot be used in real life implementations because of the limitations of the analog CNN-UM chips such as low precision or the application of non-linear templates.

Emulated digital CNN-UM architectures seem to be more flexible than their analog counterparts both in cell array size and accuracy while their computing power is just slightly smaller. By implementing these architectures on reconfigurable chips it is possible to change the cell model and evaluate the new architecture in very short time.

## II. The Ocean Model

Building a universal ocean model that can accurately describe the state of the ocean on all spatial and temporal scales is very difficult [4]. Thus ocean modeling efforts can be diversified into different classes, some concerned only with the turbulent surface boundary layers, some with continental shelves and some with the circulation in the whole ocean basin. Fine resolution models can be used to provide real-time weather forecasts for several weeks. These forecasts are very important to the fishing industry, ship routing and search and rescue operations. The more coarse resolution models are very efficient in long term global climate simulations such as simulating El Nino effects of the Pacific Ocean.

In general, ocean models describe the response of the variable density ocean to atmospheric momentum and heat forcing. In the simplest barotropic ocean model a region of the ocean's water column is vertically integrated to obtain one value for the vertically different horizontal currents. The more accurate models use several horizontal layers to describe the motion in the deeper regions of the ocean. Though these models are more accurate investigation of the barotropic ocean model is not worthless because it is relatively simple, easy to implement and it provides a good basis for the more sophisticated 3-D layered models.

The governing equations of the barotropic ocean model on a rotating Earth can be derived from the Navier-Stokes equations of incompressible fluids. Using Cartesian coordinates these equations have the following form [4], [5]:

$$\frac{du_x}{dt} = \underbrace{2\Omega \sin\theta u_y}_{Coriolis} \underbrace{- gH\frac{\partial \eta}{\partial x}}_{Pressure} + \underbrace{\tau_{wx} - \tau_{bx}}_{Lateral} + \underbrace{A\nabla^2 u_x}_{viscosity} \underbrace{- \frac{u_x}{H}\frac{\partial u_x}{\partial x} - \frac{u_y}{H}\frac{\partial u_x}{\partial y}}_{Advection} \quad (1)$$

$$\frac{du_y}{dt} = \underbrace{-2\Omega \sin\theta u_x}_{Coriolis} \underbrace{- gH\frac{\partial \eta}{\partial y}}_{Pressure} + \underbrace{\tau_{wy} - \tau_{by}}_{Lateral} + \underbrace{A\nabla^2 u_y}_{viscosity} \underbrace{- \frac{u_x}{H}\frac{\partial u_y}{\partial x} - \frac{u_y}{H}\frac{\partial u_y}{\partial y}}_{Advection} \quad (2)$$

[+] Also affiliated to Analogic and Neural Computing Laboratory, Computer and Automation Institute of HAS, Kende u. 13-17. H-1111 Budapest, Hungary.

0-7803-8359-1/04/$20.00 ©2004 IEEE

$$\frac{d\eta}{dt} = -\frac{\partial u_x}{\partial x} - \frac{\partial u_y}{\partial y} \tag{3}$$

Where $\eta$ is the height above mean sea level, $u_x$ and $u_y$ are volume transports in the x and y directions respectively. In the Coriolis term $\Omega$ is the angular rotation of the Earth and $\theta$ is the latitude. The pressure term contains H(x,y), the depth of the ocean and the gravitational acceleration g. The wind and bottom stress components in both x and y directions are represented by $\tau_{wx}$, $\tau_{wy}$, $\tau_{bx}$ and $\tau_{by}$ respectively. The lateral viscosity is denoted by A.

The bottom stress components can be linearly approximated by multiplying the $u_x$ and $u_y$ by a constant value $\sigma'$ the recommended value of this parameter is in the range $1\text{-}5\cdot10^{-7}$. The wind stress components can be computed from the wind speed above the sea surface by the following approximation [6]:

$$\tau_w = \rho C_d U_{10}^2 \tag{4}$$

Where $\rho$ is the density of the air, $U_{10}$ is the speed of the wind at 10 meters above the surface and $C_d$ is the drag coefficient. One possible approximation of the drag coefficient is the following:

$$1000C_d = 0.29 + \frac{3.1}{U_{10}} + \frac{7.7}{U_{10}^2} \quad (3 \le U_{10} \le 6\ m/s) \tag{5}$$

$$1000C_d = 0.5 + 0.071 \cdot U_{10} \quad (6 \le U_{10} \le 26\ m/s) \tag{6}$$

The horizontal friction parameter A can be computed from the mesh-box Reynolds number $R_c$:

$$R_c = \frac{\Delta x U}{A} \tag{7}$$

Where $\Delta x$ is the mesh size and U is the magnitude of the velocity in the mesh-box. By approximating U with $\sqrt{gH}$ and setting $R_c$=4 which is generally considered in nonlinear flow simulations the lateral friction can be computed by the following equation:

$$A = \frac{\Delta x \sqrt{gH}}{R_c} \tag{8}$$

At the edges of the model closed boundary conditions are used e.g. there is no mass transport across the boundaries. In this case $u_x$ and $u_y$ are both zero at the edges of the CNN cell array.

The circulation in the barotropic ocean is generally the result of the wind stress at the ocean's surface and the source sink mass flows at the basin boundaries. In this paper we use steady wind to force our model. In this case the ocean will generally arrive at a steady circulation after an initial transient behavior.

## III. CNN-UM SOLUTION

Solution of equations (1)-(3) on a CNN-UM architecture requires finite difference approximation on a uniform square grid. The spatial derivatives can be approximated by the following well known finite difference schemes and CNN templates:

$$\frac{\partial}{\partial x} \approx \frac{1}{2\Delta x}\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} = A_{dx} \tag{9}$$

$$\frac{\partial}{\partial y} \approx \frac{1}{2\Delta x}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} = A_{dy} \tag{10}$$

$$\nabla^2 \approx \frac{1}{\Delta x^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = A_n \tag{11}$$

Using these templates the pressure and lateral viscosity terms can be easily computed on a CNN-UM architecture. However the computation of the advection terms requires the following non-linear CNN template which can not be implemented on the present analog CNN-UM architectures.

$$u_{x,ij}\frac{\partial}{\partial x} \approx \frac{u_{x,ij}}{2\Delta x}\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} = A_{x,x,ij} \tag{12}$$

Most ocean models arrange the time dependent variables $u_x$, $u_y$ and $\eta$ on a staggered grid called C-grid. In this case the pressure p and height H variables are located at the center of the mesh boxes, and mass transports $u_x$ and $u_y$ at the center of the box boundaries facing the x and y directions respectively. In this case the state equation of the ocean model can be solved by a one layer CNN but the required template size is 5×5 and space variant templates should be used. Another approach to use 3 layers for the 3 time dependent variables. In this case the CNN-UM solution can be described by the following equations:

$$\frac{du_{x,ij}}{dt} = f_{ij}u_{y,ij} - gH_{ij}\sum A_{dx}\eta + \tau_{wx,ij} - \sigma' u_{x,ij} + A_{ij}\sum A_n u_x - \frac{1}{H_{ij}}\left(\sum A_{x,x,ij}u_x + \sum A_{x,y,ij}u_y\right) \tag{13}$$

$$\frac{du_{y,ij}}{dt} = -f_{ij}u_{x,ij} - gH_{ij}\sum A_{dy}\eta + \tau_{wy,ij} - \sigma' u_{y,ij} + A_{ij}\sum A_n u_y - \frac{1}{H_{ij}}\left(\sum A_{y,x,ij}u_x + \sum A_{y,y,ij}u_y\right) \tag{14}$$

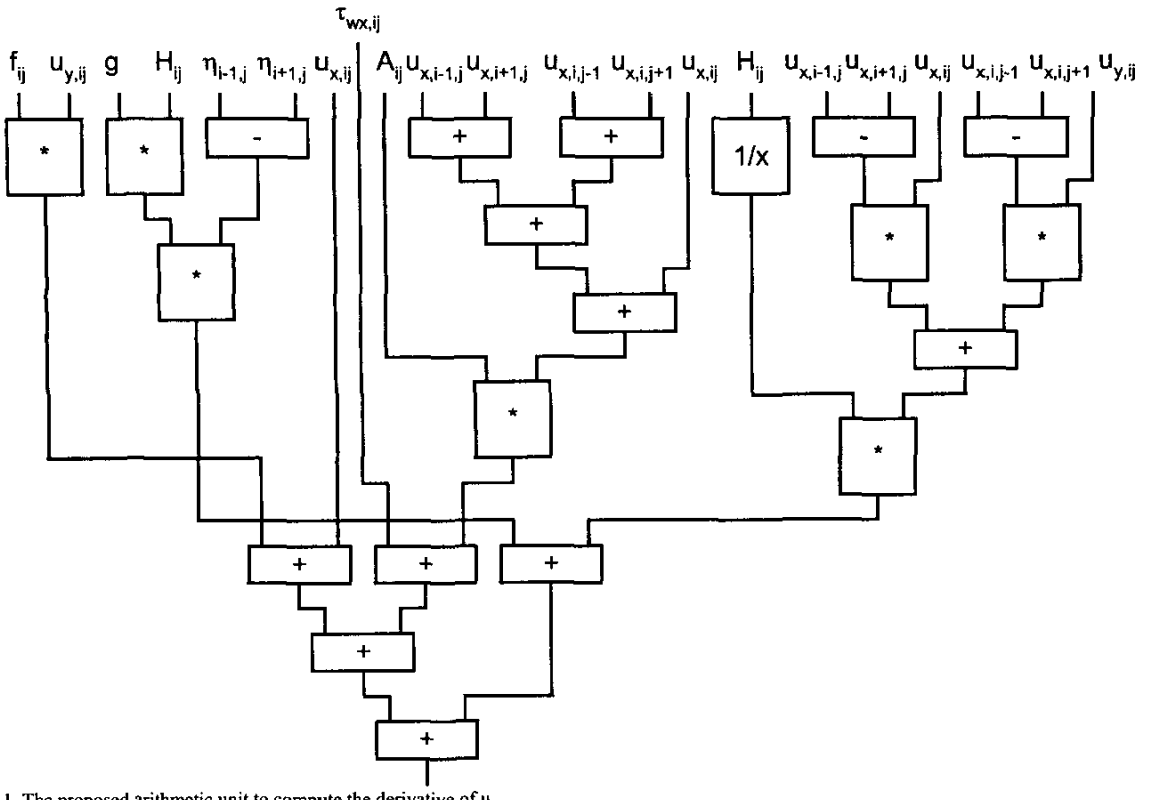


Fig. 1. The proposed arithmetic unit to compute the derivative of $u_x$.
(Multiplication with -4 in template $A_n$ and division with $2\Delta x$ and $\Delta x^2$ is done by shifts. For simplicity these shifts are not shown on the figure.)

$$\frac{d\eta_{ij}}{dt} = -\sum A_{dx} u_x - \sum A_{dy} u_y \qquad (15)$$

## IV. The Emulated Digital Solution

Using equation (13)-(15) and templates (9)-(12) an analogic algorithm can be constructed to solve the state equation of the barotropic ocean model. However the non-linear advection terms do not allow us to implement our algorithm on the present analog CNN-UM chips. The non-linear behavior can be modeled by using software simulation but this solution does not differ from the traditional approach and does not provide any performance advantage.

Some previous results show that configurable emulated digital architectures can be very efficiently used in the computation of the CNN dynamics [8] and in the solution of simple PDEs [7]. The Falcon emulated digital CNN-UM architecture can be modified to handle the non-linear templates required in the advection terms of the ocean model. The required blocks are an additional multiplier to compute the non-linearity and a memory unit to store the required $u_{x,ij}$ or $u_{y,ij}$ values. Of course this modification requires the redesign of the whole control unit of the processor.

However this modified Falcon architecture can run the analogic algorithm of the ocean model its performance would not be significant because six templates should be run to compute the next step of the ocean model. The performance can be greatly improved by designing a specialized arithmetic unit which can compute these templates fully parallel. Instead building a general CNN-UM architecture which can handle non-linear templates an array of specialized cells is designed which can directly solve the state equation of the discretized ocean model.

To emulate the behavior of the specialized cells the continuous state equations (13)-(15) must be discretized in time. In the solution the simple forward Euler formula is used but in this case we have an upper limit on the $\Delta t$ timestep. The maximal value of the timestep can be computed by using the Courant-Friedrichs-Levy (CFL) stability condition.

$$\Delta t < \Delta x / c_w \qquad (16)$$

Where $\Delta t$ is the timestep, $\Delta x$ is the distance between the grid points and $c_w$ is the speed of the surface gravity waves typically $c_w = \sqrt{gH}$ .

Computation of the derivatives of $u_x$ and $u_y$ is the most complicated part of the arithmetic unit. The proposed structure to compute the derivative of $u_x$ is shown on Fig. 1. Similar circuit is required to compute the derivative of $u_y$ fortunately the results of the multiplication of g and $H_{ij}$ and the computation of the reciprocal of $H_{ij}$ can be used in this part too. This complex arithmetic unit can compute the derivatives and update the cell's state value in one clock cycle but it requires pipelining to achieve high clock speeds. The values of $\Delta x$ and $\Delta t$ is restricted to be integer powers of two. In this case multiplication and division by these values can be done by shifts. This simple trick makes it possible to eliminate several multipliers from the arithmetic unit and greatly reduces the area requirements.

The arithmetic unit requires 20 input values in each clock cycle to compute a new cell value. It is obvious that these values cannot be provided from a separate memory. To solve this I/O bottleneck a belt of the cell array should be stored on the chip [9]. In the case of $u_x$, $u_y$ and $\eta$ two lines should be stored because these lines are required in the computation of the spatial derivatives. From the remaining values such as f, H, A, $\tau_{wx}$ and $\tau_{wy}$ only one line should be stored for synchronization purposes.

Inside the arithmetic unit fixed point number representation is used because fixed point arithmetic requires much smaller area than floating point arithmetic. The bit width of the different input values can be configured independently before the synthesis. The width of the values are computed using simple rules and heuristics.

TABLE I
RESOURCE REQUIREMENTS AND DEVICE UTILIZATION OF THE ARITHMETIC UNIT

| | Bit width | | |
|---|---|---|---|
| Variable | General | RC200 | Sim. |
| f | 18 | 10 | 9 |
| H | 17 | 10 | 13 |
| A | 17 | 10 | 8 |
| $\tau_w$ | 18 | 12 | 12 |
| $\eta$ | 34 | 30 | 20 |
| u | 31 | 30 | 20 |
| I/O bus width (bit) | 184 | 144 | 114 |
| | Required resources | | |
| 18x18 bit multiplier | 41 | 41 | |
| 18kbit Block RAM | 16 | 13 | |

| Part number | 18x18 bit multiplier utilization | | Available resources |
|---|---|---|---|
| XC2V1000 | 103% | 103% | 40 |
| XC2V8000 | 24% | 24% | 168 |
| XC2VP125 | 7% | 7% | 556 |

| Part number | 18kbit Block RAM utilization | | Available resources |
|---|---|---|---|
| XC2V1000 | 40% | 33% | 40 |
| XC2V8000 | 10% | 8% | 168 |
| XC2VP125 | 3% | 2% | 556 |

For example the deepest point of the Pacific Ocean is about 11,000 meters deep. So 14 bits is required to represent this depth H with one meter accuracy. This is far more accurate than the available data of the ocean bottom so the last or the last two bits can be safely removed. In this case we always have to multiply H by 2 or 4 if it is used in the computations. Fortunately this multiplication can be implemented by shifts. Similar considerations can be used to determine the required bit width and the displacement of the radix point for the remaining constant values such as f, A, g, $\tau_{wx}$ and $\tau_{wy}$. Using fixed point numbers with various width and displacement values makes it harder to design the arithmetic unit but it is worthwhile because the area can be reduced and clock speed is increased.

In our recent implementation predefined configurable multipliers from Xilinx are used to simplify circuit design [10]. The maximum input width of this IP core is 64 bit thus the bit width of the u and $\eta$ values can not be larger than 31 and 34 bits to avoid rounding errors inside the arithmetic unit. In this case 41 dedicated 18 bit by 18 bit signed multipliers are required to implement the arithmetic unit. Of course the bit width can be further increased by using custom multipliers. The required resources to implement this arithmetic unit on Xilinx Virtex series FPGAs is summarized in column General on Table I.

## V. RESULTS

The proposed architecture will be implemented on our RC200 prototyping board from Celoxica Ltd. [11]. The XC2V1000 FPGA on this card can host one arithmetic unit which makes it possible to compute a new cell value in one clock cycle. Unfortunately the board has 72 bit wide data bus, so 5 clock cycles are required to read a new cell value and to store the results. The performance can be increased by slightly lowering the precision as shown in column RC200 on Table I. and implementing three memory units which use the arithmetic unit alternately. In this case 4 clock cycles are required to compute 3 new cell values and the utilization of the arithmetic unit is 75%. The performance of the system is limited by the speed of the on board memory resulting in a maximum clock frequency of 90MHz. In this case the performance of the chip is 67.5 million cell update/s. The size of the memory is also a limiting factor because the input and state values must fit into the 4Mbyte memory of the board. The cell array is restricted to 512×512 cells by the limited amount of on-board memory however the XC2V1000 FPGA can be used to work with 1024 or even 2048 cell wide arrays.

By using the new Virtex-II Pro devices with larger and faster memory the performance of the architecture can reach 200MHz clock rate and can compute a new cell value in

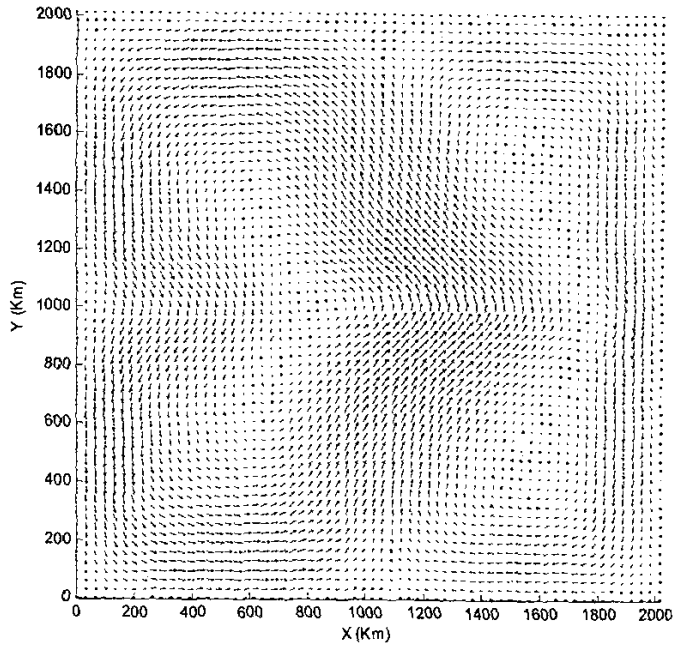


Fig. 2. Currents in the simulated ocean after 48 hours.

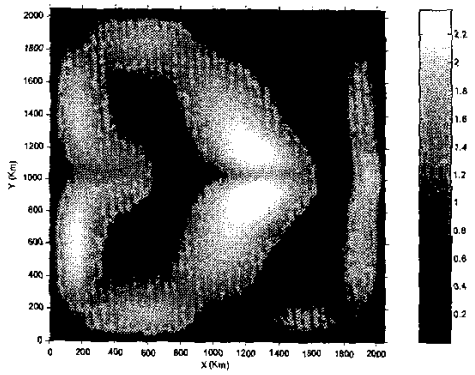

Fig. 3. Absolute value of the mass transport

each clock cycle. Additionally the huge amount of on-chip memory and multipliers on the largest XC2VP125 FPGA makes it possible to implement 14 separate arithmetic units. These arithmetic units work in parallel and the cumulative performance is 2800 million cell update/s. On the other hand the large number of arithmetic units makes it possible to implement more accurate numerical methods.

Before implementation the accuracy of the arithmetic unit is determined by simulation. To avoid rounding errors inside the arithmetic unit the width of the partial results can not be larger than 64 bits. Simulation using larger accuracy requires the use of fixed point library such as provided in the SystemC package [12] which results in very low performance. The bit width of the variables in this case is shown on Table I.

The results of this fixed point computation is compared to the 64 bit floating point results. To evaluate our solution a simple model is used. The size of the modeled ocean is 2097km, the boundaries are closed, the grid size is 512×512 and the grid resolution is 4096m. The ocean bottom is divided into two parts; in the upper half the depth is 2000m while the lower half is 4000m deep. The model is forced by a wind blowing from south and its value is constant along the y direction. The wind speed in the x direction is described by a reversed parabolic curve where the speed is zero at the edges and 8m/s in the center.

The fixed point results after 48 hours of simulation time with 16s timestep is shown on Fig. 2. and 3. The computation time was 56 minutes on an Athlon XP 1800+ processor. This is equivalent to approximately 0.85 million cell update/s. The absolute value of the difference between the floating point and the fixed point computation is shown on Fig. 4. and 5. In spite of the small precision of the simulated arithmetic unit the results are quite good the accuracy is about 0.6% if compared to the maximum value of the flow. The implemented arithmetic unit with larger bit width will achieve even better accuracy.

The performance of the emulated digital solution of the model is very encouraging even using the mid-sized XC2V1000 FPGA on our prototyping board the previous computation takes just 41s thus the computation is accelerated by 81 times. Using the currently available largest FPGA the XC2VP125 the speedup is 3300-fold; in this case the whole computation takes just 1 second!

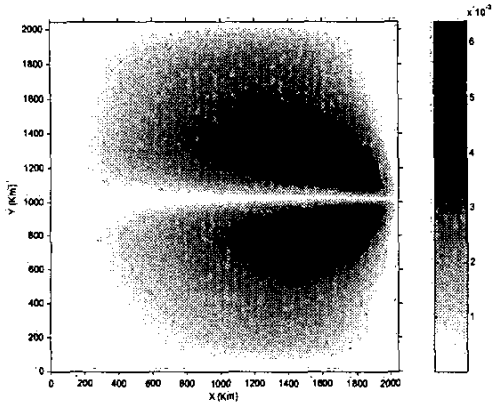

Fig. 4. Absolute difference between the fixed point and the floating point solution of $u_x$.

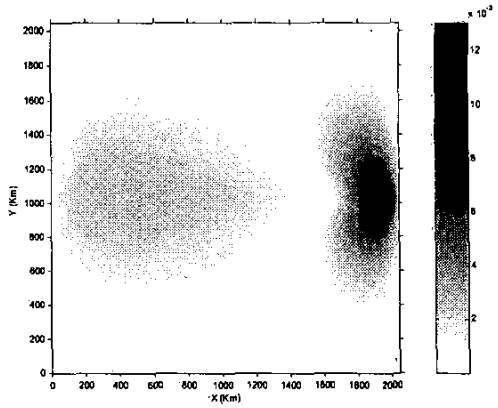

Fig. 5. Absolute difference between the fixed point and the floating point solution of $u_y$.

## VI. Conclusions

The state equation of a barotropic ocean model with realistic input parameters was solved using a CNN architecture. The cell model of the Falcon emulated digital CNN-UM processor was modified according to the governing equations of the ocean model. The CNN-UM solution can be efficiently accelerated using re-configure devices. The proposed architecture was implemented on a mid-sized FPGA with one million equivalent system gates on our RC200 prototyping board. This solution is 80 times faster than an Athlon XP 1800+ processor while using larger FPGA and more memory 3300-fold performance increase can be achieved. In spite of the low precision used in the arithmetic unit the results are very close to the 64 bit floating point computations.

The available resources on the current high performance FPGAs makes it possible to implement higher order numerical methods to solve the state equation of the ocean model. On the other hand the more sophisticated 3-D layered models also can be implemented on these devices.

## References


[1] T. Roska and L. O. Chua, "The CNN Universal Machine. An analogic array computer", *IEEE Trans. On Circuits and Systems-II*, Vol.40, pp. 163-173, 1993.

[2] T. Roska, T. Kozek, D. Wolf, L. O. Chua: "Solving Partial Differential Equations by CNN" *Proc. of European Conf. on Circuits Theory and Design* 1992

[3] P. Szolgay, G. Vörös, Gy. Erőss "On the Applications of the Cellular Neural Network Paradigm in Mechanical Vibrating System", *IEEE. Trans. Circuits and Systems-I, Fundamental Theory and Appl.*, vol. 40, no. 3, pp. 222-227, 1993.

[4] L. Kantha, S. Piacsek, "Ocean Models" [Online]. Available: http://csep1.phy.ornl.gov/CSEP/OM/OM.html

[5] R. Robertson, L. Padman, G. D. Egbert, "Tides in the Weddell Sea," 1998,[Online]. Available: http://www.esr.org/antarctic/barotropic.html

[6] R. H. Stewart "Introduction To Physical Oceanography", 2003, [Online]. Available: http://oceanworld.tamu.edu/resources/ocng_textbook/contents.html

[7] Z. Nagy, P. Szolgay "Numerical solution of a class of PDEs by using emulated digital CNN-UM on FPGAs" in *Proc. of 16th European Conf. on Circuits Theory and Design*, Cracow, September 1-4, 2003

[8] Z. Nagy, P. Szolgay "Configurable Multi-Layer CNN-UM Emulator on FPGA" *IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications*, Vol. 50, pp. 774-778, 2003

[9] P. Keresztes, Á. Zarándy, T. Roska, P. Szolgay, T. Hídvégi, Péter Jónás, A. Katona: "An emulated digital CNN implementation", *International Journal of VLSI Signal Processing*, Kluwer, 1999 September 9.

[10] Xilinx Products Homepage [Online]. Available: http://www.xilinx.com

[11] Celoxica Ltd. Homepage [Online]. Available: http://www.celoxica.com

[12] Open SystemC™ Initiative Homepage [Online]. Available: http://www.systemc.org