High performance reconfigurable computing with cellular automata

Murtaza, S.

Publication date
2010

Chapter 8

Performance Predictions of CA Computations on Advanced FPGA Computing Resources

Performance modeling and evaluation of FPGA based CA accelerators have been the main focus of this research, with real number based lattice Boltzmann computations as the prime candidate. Real number based computations are not only FPGA logic resource hungry but also place constraints on the data transfer interface between the FPGA and the external components like memory banks. Confining such a resource hungry application within the available FPGA resources is always a challenge. Starting from a single to multiple FPGA based implementations, previous chapters have demonstrated the successful implementations of our proposed performance model. Being aware of the strengths and weaknesses of our model, in this chapter, we take the liberty of using the model to predict CA implementations using advanced computing resources and CA algorithms. In the absence of advanced computing resources, pencil and paper based computations are always an option. This chapter focuses on performance predictions for advanced computing resources and CA algorithms.

8.1 FPGA with 61 PEs

A single FPGA based accelerator as presented in Chapter 5, is either IO bound or compute bound for a given set of parameters as specified by Equation (3.9). And these values are defined by the available FPGA device specification. For example, updating a single LBM cell PE requires 674 FPGA core clock-cycles, a number of cells read by the FPGA in parallel from the attached on-board memory bank $k=\frac{1}{10}$, on an average the FPGA takes 1.1 core clock-cycles to read a single word (64-bits) from a source memory bank, and design implementations on a FPGA board with memory banks run at 150MHZ and

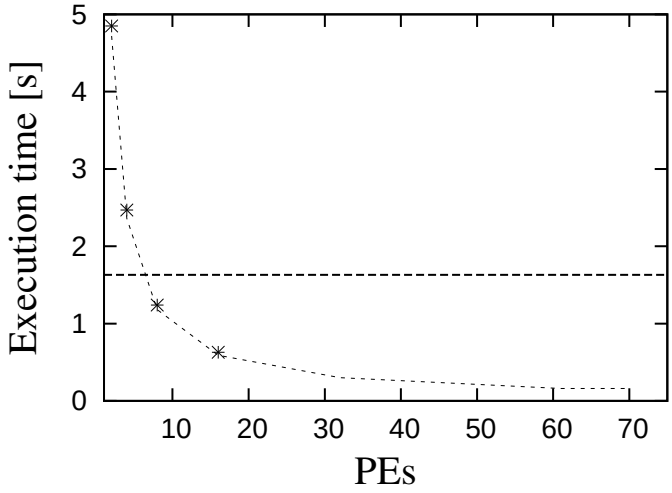

Figure 8.1: *Single FPGA based performance prediction for computing 512 iterations of the 64x64 LBM D2Q9 grid. Stars represent the measurements similar to as shown in Figure 5.6 and curve is the model. Broken line represents the Fortran implementation.*

180MHz respectively. For these values, the accelerator stays compute bound up to a maximum of 61 PEs, however, the available logic using the FX140 FPGA only allowed a maximum of 16 PE implementation. In this section we look into the performance of FPGA based D2Q9 LBM implementations using the maximum number of PEs, that is, 61.

8.1.1 Single FPGA based D2Q9 LBM Implementation

Similar to Figure 5.6, a performance prediction for a single FPGA based implementation, extended to a maximum possible of 61 PE implementations is shown in Figure 8.1. FX140 chip resources allowed a maximum of 16 PE implementation which achieved a speedup of 2.3 as compared with the Fortran implementation. If the chip resources could accommodate 61 PE implementation, the performance would further improve by a factor close to four and also by an order of speedup as compared to Fortran implementation. Beyond 61 PEs, implementation is no longer compute bound and therefore execution time stays constant with overall decrease in the efficiency.

As shown in Figure 8.1, a 64x64 D2Q9 lattice computed for 512 iterations using a 61 PEs implementation, reaches a theoretical peak performance of 13.10 MLUPS (million lattice-site updates per second).

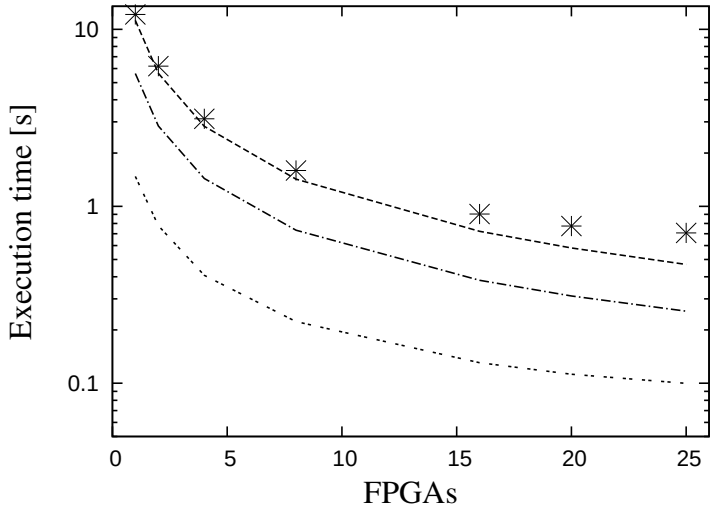

Figure 8.2: *FPGA enabled PC cluster based performance prediction for computing 10 iterations of the 1000x2000 D2Q9 LBM grid. Top most broken curve represents the model and the stars are the measurements as shown in Figure 7.4 for an 8PE per FPGA implementation. The middle and bottom curves are the model for a 16PE and 61PE per FPGA implementation respectively.*

8.1.2 FPGA enabled PC Cluster Based D2Q9 LBM Implementation

Multiple FPGA based implementations were run using the Maxwell machine which includes FX100 FPGAs. Since the available logic resources on FX100 are less when compared to FX140 FPGA, therefore, implementations tested on Maxwell allowed a maximum of 8 PEs per FPGA. Had Maxwell's FPGAs been FX140, then a 16 PE per FPGA implementation would have been possible (same as our single FPGA based implementations using FX140, that allowed a maximum of 16 PE implementation), and a theoretical maximum of 61 PEs per FPGA using advanced chips of the future. Based on the three specified flavors of FPGA hardware, performance for the multiple FPGA enabled PC cluster implementation as a function of FPGAs for a given system size of 1000x2000 cells is as shown in Figure 8.2: top most curve represents the 8 PEs per FX100 FPGA implementation; middle curve represents the 16 PE per FX140 FPGA implementation that would improve the performance by a factor of 2; and the bottom curve represents the theoretical maximum of 61 PEs per FPGA that would further improve the overall system performance by a factor of 4.7 for a 25 FPGAs based setup.

As shown in Figure 8.2, a 2 million D2Q9 lattice computed for 10 iterations, using 25 FPGA implementation each with 61 PEs, reaches a theoretical peak performance of 200.4 MLUPS.

8.2 3D LBM Performance Prediction

For a 3D LBM implementation, let us consider the D3Q19 lattice Boltzmann model. The number of floating points to represent a D3Q19 cell, and the required number of operations to update this cell, almost double when compared to that of a D2Q9 cell. To represent a cell state for a D3Q19 model using our D2Q9 based implementation strategy, requires twenty 64-bit numbers (18 for a given cells neighbours, another representing the particle at rest and an extra redundant one similar to D2Q9 implementation). This implies $k=\frac{1}{20}$, 1350 FPGA core clock-ticks to update a cell (D2Q9 cell update takes 674 clock-ticks). A single PE implementation for a D3Q19 model would remain more or less similar to that of a D2Q9 implementation as shown in Figure 5.2, however, the size of the register bank needs to be larger and the state machine that drives the PE data path should have comparatively more states. In terms of the hardware resources, the main compute core PE remains same as D2Q9 model, that is, floating point cores for sum, multiplication and division remain the same and rest can be ignored for the purpose of simplicity. Therefore, a single FPGA based implementation for the above stated numbers, stays compute bound for a maximum of 61 PEs using Equation (3.9), the same as D2Q9 implementation. In the following section we look into the performance prediction of such systems.

8.2.1 Single FPGA based Implementation

For a single FPGA based D3Q19 model implementation, the execution time for the compute bound case is similar to the D2Q9 model as specified by Equation (3.10). Using the given performance model, we predict the execution time for varying system sizes of D3Q19 model computed over 256 iterations. In the D2Q9 model implementation, we could only manage to achieve 16 PEs on a FX140 chip, though theoretically the system can include 61 PEs to stay compute bound. Based on this, we have predicted the performance model for the D3Q19 implementation. Figure 8.3(left), represents the execution time as a function of system size for a D2Q9 model computed over 256 iterations using 16 and 61 PEs respectively. Similarly, the D3Q19 model is as shown on the right-hand-side of the same figure. The figures demonstrate that for a particular system size, the execution time for D3Q19 model is almost double to that of the D2Q9 model implementation.

As shown in Figure 8.3(left), a 262144 cell D2Q9 lattice computed for 256 iterations using a 61 PEs implementation, reaches a theoretical peak performance of 13.55 MLUPS. And a similar sized D3Q19 lattice computed for 256 iterations using a 61 PEs implementation, reaches a theoretical peak performance of 6.77 MLUPS.

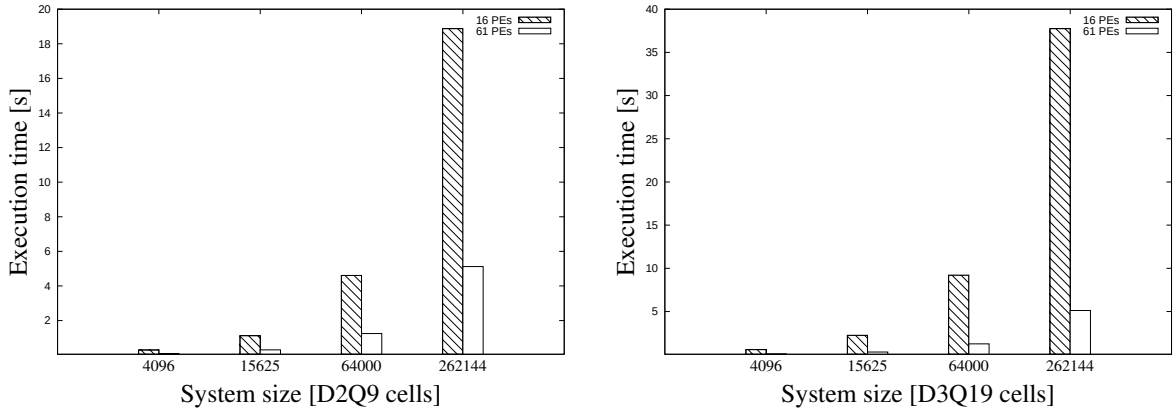

Figure 8.3: *Performance prediction comparison for a FPGA-enabled-PC based D2Q9 and D3Q19 LBM model. Execution time as a function of system size for a single FPGA based implementation. (Left) Each data point represents the execution time to compute 256 iterations of a D2Q9 LBM grid for a given system size with 16 and 61PEs respectively. (Right) Each data point represents the execution time to compute 256 iterations of a D3Q19 LBM grid for a given system size using 16 and 61 PEs respectively.*

8.2.2 FPGA enabled PC Cluster Implementation

For a FPGA enabled PC cluster based D3Q19 implementation, we consider a 3D rectangular problem domain of size 2.04 million cells, decomposed along the longest dimension into F equal sized chunks (1-D domain decomposition setup similar to the 2D implementation as explained in Chapter 7) and computed for 256 iterations using F FPGAs. Each iteration computation is a three-step process similar to the D2Q9 implementation strategy as explained in Chapter 7, that is, starting with boundary cells, followed by bulk cells and finally by the on-board memory update (see Section 7.1 for details). However, the boundary processing, bulk processing, and the boundary download, as specified in Section 7.1, for a D2Q9 implementation differs from D3Q19 implementation as shown below:

8.2.2.1 Boundary Processing

Each FPGA starts with computing boundary cells and can be specified as

$$T_{\mathrm{b}} \;=\; \tau_{\mathrm{r}} + \left\{ \frac{2xz}{p} \right\} \tau_{\mathrm{c}} \tag{8.1}$$

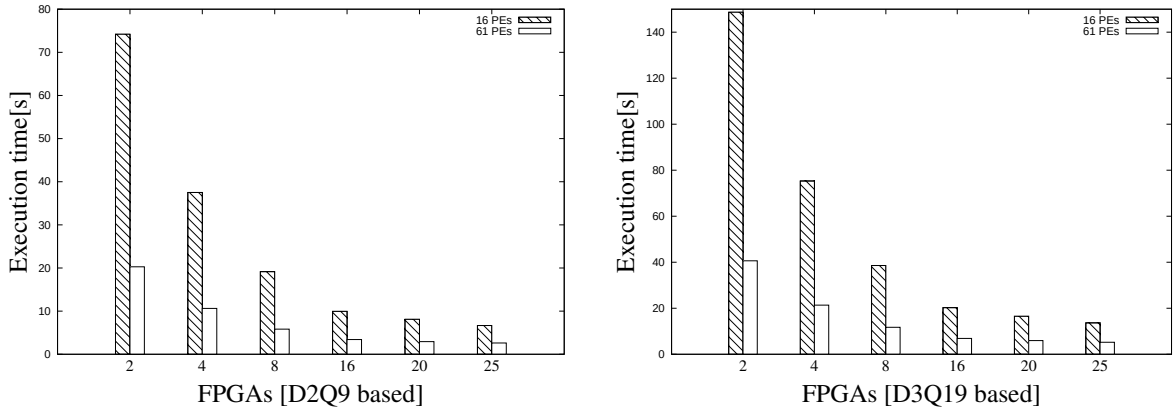

Figure 8.4: *Performance prediction comparison for a FPGA-enabled-cluster based D2Q9 and D3Q19 LBM model. Execution time as a function of FPGAs for a FPGA enabled PC cluster implementation. (Left) Each data point represents the execution time to compute 256 iterations of a 2.04 million cells D2Q9 LBM grid using 16 and 61PEs respectively. (Right) Each data point represents the execution time to compute 256 iterations of a 2.04 million cells D3Q19 LBM grid using 16 and 61 PEs respectively.*

8.2.2.2 Bulk Processing

After the boundary cells, each FPGA computes the bulk cells and in parallel the respective host is informed to start the boundary cells update process around the neighbouring FPGAs. The time to compute the bulk cells and to update the boundary cells is specified as

$$T_{\mathrm{k}} = \frac{1}{p}\left\{\frac{N}{F} - 2xz\right\}\tau_{\mathrm{c}} + \left\{\frac{p}{k} - 1\right\}\tau_{\mathrm{r}} + \tau_{\mathrm{w}} \tag{8.2}$$

$$T_{\mathrm{u}} = 2xz\left\{\tau_{\mathrm{d}} + \tau_{\mathrm{b}} + \tau_{\mathrm{m}}\right\} \tag{8.3}$$

8.2.2.3 Boundary Download

Finally upon the completion of the next iteration computation, the host process downloads the updated boundary data to its connected FPGA modules source bank. The time to download boundary cells (T_{d}) is $2xz\tau_{\mathrm{d}}$. The sum of the above mentioned time durations results in the overall execution time for the FPGA enabled PC cluster implementation.

$$T_{\mathrm{F}} = T_{\mathrm{b}} + \max\left\{T_{\mathrm{u}}, T_{\mathrm{k}}\right\} + T_{\mathrm{d}}. \tag{8.4}$$

Comparing the execution times of D2Q9 and the D3Q19 model as a function of FPGAs as shown in Figure 8.4 respectively, we notice, a system size of 2.04 million cells computed

over 256 generations using 16 and 61PE respectively. The three dimensional setup takes twice the time as compared to the two dimensional setup.

As shown in Figure 8.4, a 2.04 million D2Q9 lattice computed for 256 iterations, using 25 FPGA implementation each with 61 PEs, reaches a theoretical peak performance of 200.23 MLUPS. And a similar sized D3Q19 lattice computed for 256 iterations, using 25 FPGA implementation each with 61 PEs, reaches a theoretical peak performance of 100.11 MLUPS.

8.3 Summary

Using pencil and paper approach we deduced the theoretical limits of our model implementations and their performance relative to the existing implementations. For a single FPGA based implementation, improving the number of PEs from 16 to a theoretical maximum of 61, improves the overall performance by a factor close to four. A theoretical peak performance of 13 MLUPS is expected for such system implementations. For a multiple FPGA enabled PC cluster implementation, improving the number of PEs per FPGA from eight to a theoretical maximum of 61, improves the overall performance of D2Q9 implementation by a factor close to five, with an expected theoretical peak performance of 200 MLUPS. Secondly, by porting our two-dimensional performance model to a three-dimensional D3Q19 LBM implementation, a linear increment in terms of execution time is identified. FPGA logic resources required for D3Q19 PE implementation would largely resemble D2Q9 PE in terms of floating point units and control logic, though a larger register file would be required to store and compute the higher number of computations.