Virtualizing Analog Mesh Computers: The Case of a Photonic PDE Solving Accelerator

Jeff Anderson
Department of Electrical and Computer Engineering
The George Washington University
Washington, DC 20052
Email: jeffa@gwu.edu

Engin Kayraklioglu
Cray, Inc.
Email: engin@gwu.edu

Hamid Reza Imani
Department of Electrical and Computer Engineering
The George Washington University
Washington, DC 20052
Email: hamid@gwu.edu

Mario Miscuglio
Department of Electrical and Computer Engineering
The George Washington University
Washington, DC 20052
Email: mmiscuglio@gwu.edu

Volker J. Sorger
Department of Electrical and Computer Engineering
The George Washington University
Washington, DC 20052
Email: sorger@gwu.edu

Tarek El-Ghazawi
Department of Electrical and Computer Engineering
The George Washington University
Washington, DC 20052
Email: tarek@gwu.edu

2020 International Conference on Rebooting Computing (ICRC) | 978-1-6654-1975-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICRC2020.2020.00008

***Abstract*—Innovative processor architectures play a critical role in sustaining performance improvements under severe limitations imposed by feature size and energy consumption. The Reconfigurable Optical Computer (ROC) is one such innovative, Post-Moore's Law processor. ROC is designed to solve partial differential equations in one shot as opposed to existing solutions, which are based on costly iterative computations. This is achieved by leveraging physical properties of a mesh of optical components that behave similarly to electrical resistances. However, building large photonic arrays to accommodate arbitrarily large problems is not yet feasible. It is also possible to have problems that are smaller than the size of the accelerator array. In both cases, virtualization is necessary. In this work, we introduce an architecture and methodology for light-weight virtualization of ROC. We show that overhead from virtualization is minimal, and our experimental results show two orders of magnitude increased speed as compared to microprocessor execution while keeping errors due to virtualization under 10%.**

***Keywords*-Photonic computing; Hardware acceleration; Analog computers; Virtualization; Scientific computing; Emerging Technology; Parallel Processing**

I. Introduction

Problems associated with extreme feature reduction, particularly the end of Moore's law and Dennard scaling, have introduced operational limits in both clock speed and levels of energy consumption. For large-scale simulations, the standard approach uses supercomputers comprising chips with tens of cores, and accelerated with digital co-processors such as Intel Xeon Phi KNL, graphics processing units (GPU) and field programmable gate arrays (FPGA). A recent example of this is the SW26010 processor with its on-chip manycore accelerator. Such digital supercomputers can be very power hungry, such as the Sunway TaihuLight, which consumes 15MW [1]. This translates roughly to $15M annually in operating costs. A departure from conventional hardware architectures and technology is becoming necessary.

Historically, analog computers have been used where digital computers could not provide timely solutions; not only in real-time control, but also in modeling and simulations. One prominent methodology is to model dynamic physical systems using analog electric circuits, implemented with either discrete components or as an application-specific integrated circuit (ASIC), hosted by a digital computer which regulates access to the accelerator.

Many computational problems in science and engineering are modeled via solving partial differential equations (PDEs) [2]. These equations are commonly solved by first discretizing them into finite difference equations or finite elements. Iterative methods such as conjugate gradient or adaptive multi-grid are then adopted to solve these equations [3]. Due to the large number of elements, as well as the iterative nature of obtaining the solution, such computations are processor-intensive. Illustrated in Figure 1, electrical circuit analogs of PDEs drastically reduce computation time by providing a solution in one shot [4], [5]. These electric analog mesh computers require a minimum computation time dependent on the lumped resistance and capacitance values of the entire network. However, recent advances in material science have enabled nanophotonic implementations of analog mesh computers which address this shortcoming by reducing computation time to the time-of-flight of an optical signal through the mesh diameter [6], [7].

As the sizes of physical simulations continue to scale higher, they will inevitably be limited by hardware resources. This is a common problem in classical computing and has

978-1-6654-1975-8/20/$31.00 ©2020 IEEE
DOI 10.1109/ICRC2020.2020.00008

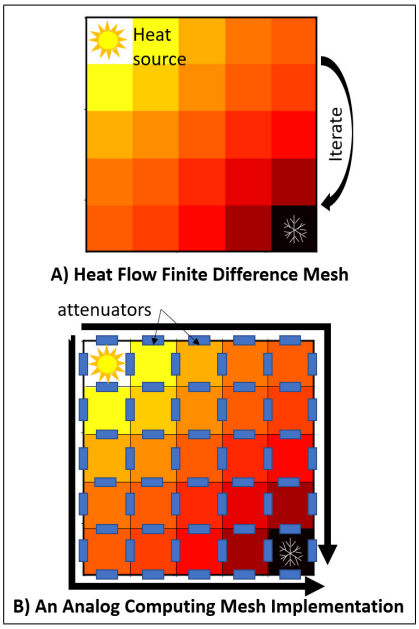

Figure 1: Mapping a steady-state heat flow PDE to a finite difference mesh and an analog mesh computer.

historically been solved using virtualization [8]. Traditional software virtualization methods, such as paravirtualization, require an operating system (OS) to coordinate with the underlying hypervisor [9]. Virtualization techniques have since moved to lower levels of the software stack when applied to reconfigurable accelerators [10] and heterogeneous systems [11]. However, workload asymmetries expected in multi-user environments benefit more from virtualization at the hardware level, enabling space-time sharing of the reconfigurable system [12], [13].

In this paper, we present a hardware virtualization architecture for analog mesh computers and apply it to the reconfigurable optical computer (ROC) [7]. These techniques enable a fixed-size analog mesh accelerator to emulate PDEs of varying sizes, thereby supporting the expected scaling of physical simulation workloads.

II. Background and Related Work

The complexity of PDE computation has led to the introduction of a variety of acceleration methods. One such class of techniques, numerical methods, is the most common approach for solving PDEs. The three most widely used numerical methods are finite element methods (FEM), finite volume methods (FVM) and finite difference methods (FDM) [14]. These techniques discretize the problem into a computational grid (or mesh) and solve for each grid point iteratively. For these techniques, grid point spacing is inversely proportional to the accuracy of the solution. Consequently, an increase in grid resolution results in an increased time-to-solution, due to an increased number of required computational iterations. Traditional digital PDE accelerators based on complementary-metal-oxide-semiconductor (CMOS) graphics processing units (GPU) and central processing units (CPU) [15], [16], obtain speedups from clock-frequency scaling and enhanced parallelism, but are bound by the number of required iterations. Other techniques, such as adaptive mesh refinement (AMR), reduce the time-to-solution of numerical methods by exploiting distinct regions of spatially-localized features that occur in physical simulations [17]. By creating multiple meshes of varying resolutions which overlay the original mesh, AMR reduces the number of iterations with a minimal reduction in accuracy.

One extreme example of a PDE accelerator, the electrical analog mesh computer, uses a network of resistors to solve the Poisson Equation in one shot [4] and has found use in solving oscillatory flow [18] and resistivity log interpretations [19]. This eliminates the iterative component required by numerical solutions, effectively reducing the time-to-solution to the time required for a signal to traverse the network diameter. The electrical analog mesh computer exploits the relationship between difference equations (Equation 1) used in FDM and Kirchoff's Laws for voltage and current, allowing Equation 1 to be rewritten as a sum of mesh currents (Equation 2). This effectively maps the FDM computational grid to an electric mesh, where distance, h, dictates the mesh resolution, characterized as the number of resistors in the mesh. PDEs are mapped to an analog mesh by setting boundary conditions as potentials at various *stiff* nodes around the mesh perimeter, with the remaining *free* nodes responsible for grid computation. Variations of electrical analog mesh computers have been proposed to solve a variety of PDEs, as simple resistor-inductor-capacitor (RLC) configuration changes allow them to describe many physical phenomena [5].

$$\nabla^2 \vec{f} \simeq \frac{1}{h^2}\Big[\vec{f}(\vec{P_1}) + \vec{f}(\vec{P_2}) + \vec{f}(\vec{P_3}) + \vec{f}(\vec{P_4}) - 4(\vec{f}(\vec{P_0}))\Big] \tag{1}$$

$$\frac{1}{G}\Big[(I_1 - I_0) + (I_2 - I_0) + (I_3 - I_0) + (I_4 - I_0)\Big] = 0 \tag{2}$$

Shown in Figure 2, ROC was proposed as a class of reconfigurable nano-optical analog mesh computers [6]. Loosely coupled to a larger computer system, ROC is presented with digital data used to configure the mesh and the boundary. Digital results are, in turn, read from the mesh upon completion of the PDE computation. ROC replaces electric mesh components with Reconfigurable Optical Elements (ROEs), implemented with a network of silicon-photonic routers and attenuators or nano-optical metatronics [7]. Metatronics emulate combinations of RLC devices through changes in real refractive index and imaginary extinction coefficients, as governed by permittivity, which influence diffusion currents traversing the waveguide [20]. Indium tin oxide (ITO), typically employed as an electro-optic modulator, has also been

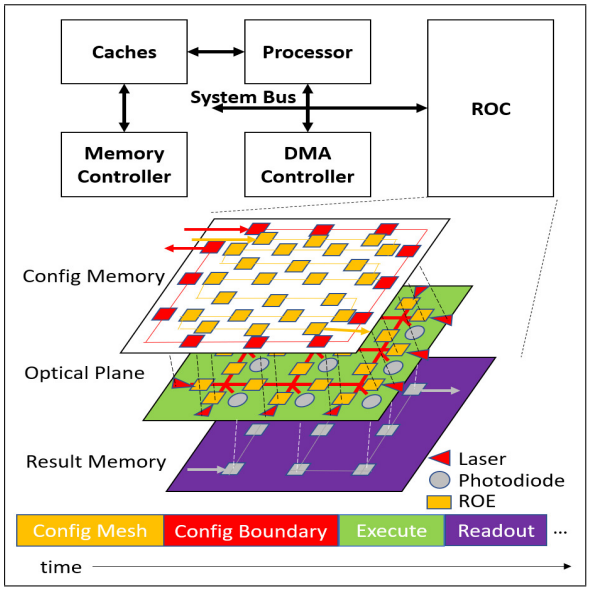

Figure 2: ROC hardware stack comprising layers supporting reconfiguration, computation and readout.

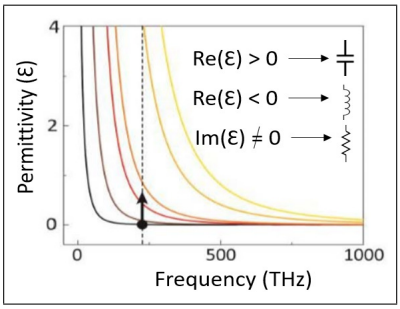

Figure 3: ITO ROEs biased to RLC equivalents [22], [23].

shown to exhibit metatronic behavior at high frequencies [21]. Illustrated in Figure 3, an ITO ROE is biased when presented with a voltage, affecting its permittivity at a given wavelength, thus its attenuation of an optical signal [22].

For the heat transfer application in Figure 1, ROC can emulate an electrical analog mesh computer by enabling multiplication and addition operations to be executed using diffusion currents. This allows conductance, G in Equation 2, to be replaced with the transmission of optical power associated with electromagnetic radiation through the network. For signal wavelengths smaller than the device feature size, addition is done via superposition of electromagnetic waves from multiple waveguides by means of optical interference. Consequently, ROEs can vary the characteristics of the surface under simulation via coordinated changes in attenuation. This provides ROC with a degree of programmability, where the optical mesh can be reconfigured to match attributes of the computational mesh (enabling common mesh acceleration techniques such as AMR), and in cases of non-rectangular shapes, to effectively turn ROEs "off" to match the shape of a surface.

Due to the fixed size of the analog mesh computer from Figure 1, any solution obtained from it will have a fixed resolution. A solution which requires a lower resolution than that supported by the mesh hardware can be computed with selective pruning of nodes. Conversely, any solution which requires a higher resolution or size than what is natively supported by hardware cannot be computed. This limits the utility of analog mesh computers in scientific programming, where computational mesh sizes are expected to scale with problem size and resolution requirements.

Similar problems encountered in the classical computing domain have traditionally been solved by virtualization [8]. More recently, virtualization has been applied to reprogrammable accelerators, where resources are coordinated without user assistance [24], [25]. Many of these techniques were derived from strategies used by operating systems to support virtual memory [12]. In these techniques, an OS or hypervisor provides a hardware abstraction layer (HAL) between hardware resources and software [26], allowing software to view the resources as an autonomously coordinated pool.

Heterogeneous systems have since virtualized FPGAs, taking advantage of their reconfigurability for specific workloads [10], [13]. These techniques also borrow from OS-based virtualization strategies, due to the coordination required for setup and scheduling [27]. However, lightweight virtualization strategies are often needed [26]. El-Araby developed a lightweight virtualization strategy which uses the concept of "virtual hardware" to support space-time sharing of reconfigurable resources [12]. Space-time sharing combats resource underutilization frequently encountered during multi-tasking operations due to asymmetric workloads [12]. Lightweight hardware virtualization techniques were also developed by Taher to enable fine-grained resource utilization and load balancing in reconfigurable, heterogeneous systems [13]. As analog mesh computers serve the same capacity as FPGA accelerators within a heterogeneous architecture, their workloads will be similar, calling for hardware virtualization techniques and space-time sharing.

III. Virtualization of an Analog Mesh

To enable virtualization, we extend the "Computing in Time - Computing in Space" paradigm [28] to analog mesh computers. We do this by introducing a hardware virtualization infrastructure which enables space, time and/or space-time sharing of hardware resources. This allows a computer with fixed hardware resources to compute a solution to a problem of a larger size than the computer was originally designed to handle.

For an analog mesh computer, this means matching a computational mesh in terms of both *size* and *resolution*. Matching the size of a computational mesh ensures that the

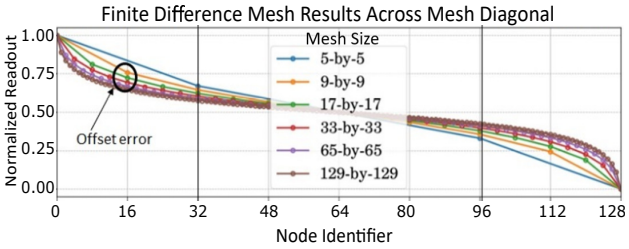

Figure 4: Error in mesh calculations (across the diagonal) as influenced by mesh resolution for the diagonal PDE configuration illustrated in Figure 1.

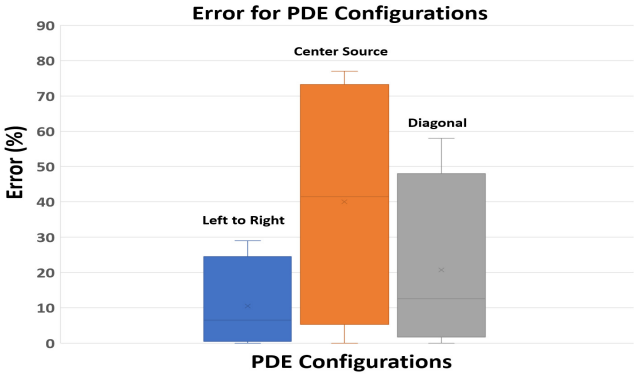

Figure 5: Error in mesh calculations for corresponding points in 4x4 and 8x8 meshes as influenced by PDE configuration.

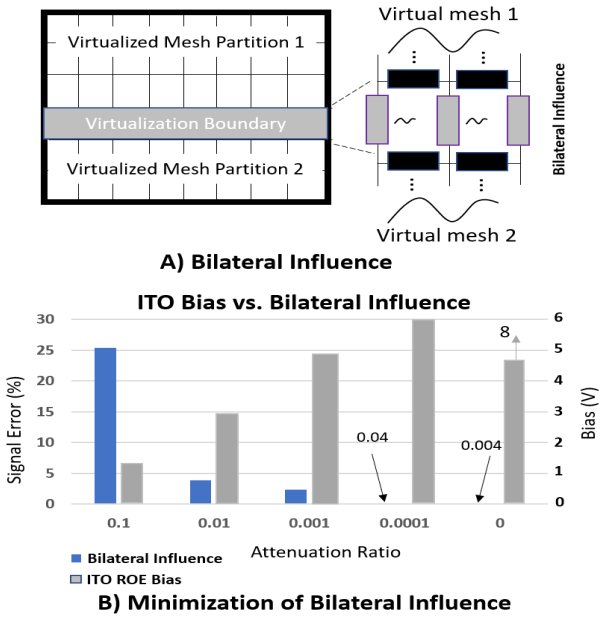

Figure 6: Virtualization Boundary theory of operation (A) and bilateral influence at varying ITO attenuation ratios (B).

final solution contains all regions within the computational mesh boundary, while matching resolution ensures that the number of nodes within the computational mesh matches the number of nodes within the physical mesh. Interestingly, an analog mesh computer can be viewed as an approximate computing engine, as the resolution, thus accuracy, of the solution is dependent on the size of the mesh. This characteristic allows computer architects to select the lowest-energy mesh that meets a given precision requirement. The case of a low-resolution computational mesh mapped to a high-resolution analog mesh computer is trivial; since the computer executes the calculations in a one-shot fashion, minimal time is wasted in computing an unnecessarily high-resolution solution and results can be pruned to match the required resolution. The case of a high-resolution computational mesh mapped to a low-resolution analog mesh computer is more interesting in that the missing information must be estimated, resulting in a form of aliasing.

Figure 4 shows the error associated with mismatches in resolution along the diagonal of the ROC optical mesh for the diagonal PDE configuration from Figure 1. The readout value for corresponding points among meshes of different sizes varies, resulting in a resolution-dependent error called *offset*. Minimum offset appears at both stiff nodes and free nodes farthest from stiff sources. Figure 5 shows our results computing error between corresponding points in multiple 4x4 and 8x8 meshes for different PDEs. The difference in error ranges, calculated as the ratio of 4x4/8x8 offset to the mesh dynamic range, implies that varying characteristics in PDE configuration (e.g. boundary conditions, etc.) influence offset.

A. *Space Sharing*

Multi-user environments require an analog mesh to compute the solution to multiple problems simultaneously. Shown in Figure 6, simultaneous computation is enabled by partitioning the mesh, where a *virtualization boundary* is created between *virtualized mesh partitions* through reprogramming of boundary attenuators to maximize loss. Signal diffusion through the virtualization boundary, called *bilateral influence*, is possible and reduces the signal-to-noise ratios (SNR) of virtualized mesh partitions, thus increasing error.

Due to the metatronic-electrical equivalence reported by Gui [21], we simulated ITO-based metatronic circuits with SPICE [29] to determine an appropriate attenuation ratio between the virtualization boundary and individual virtualized mesh partitions. Transient analysis results illustrate that a virtualization boundary having an attenuation 4 orders of magnitude larger than that within the virtualized mesh partition (simulated with a resistive divider) reduces the *virtualization error* associated with bilateral influence to less than 1%. Bias values required to attain such isolation using plasmonic MOS waveguides with a tunable ITO region were estimated from research in the field [30].

Space sharing of the analog mesh requires hardware overhead, in the form of the number of attenuators required to sustain the virtualization boundary. The overhead, calculated as the quantity of boundary attenuators divided by virtualized mesh partition attenuators, varies with virtual mesh size. This component overhead can be used to derive overhead in terms of power or size by multiplying the overhead ratio by the consumed power or size of each attenuator. Additional overhead comes in the form of latency associated with writing to attenuators during the *setup* and *teardown* of the virtualization boundary.

B. Time Sharing

Time sharing is required when the number of tasks is greater than the resource. This forces each task to share the resource and make gradual progress to a solution.

A similar situation arises when the problem size and resolution is larger than can be supported by a single analog mesh computer. When this situation arises, the problem is divided into regions, and each region must be scheduled and solved for individually. The analog mesh computer provides two techniques for supporting problem sizes larger than the physical mesh: 1) recursive mesh refinement (RMR), where the analog mesh is used exclusively to solve for each region, and 2) linear interpolation, where the analog mesh computer is used as an approximate computer for the initial computation.

1) Recursive Mesh Refinement: We propose a method to virtualize an analog mesh computer called *Recursive Mesh Refinement* (RMR), which enables a coarse-grained analog mesh computer to approximate the solution of a much finer-grained problem. The notion of starting from a coarse resolution and, through recursive refinement, arriving at an improved solution has been used in many other scenarios such as wavelet-based image registration [31]. This technique enables hardware designers to select the coarsest mesh dimension allowable by the common case, and then use the coarse mesh with RMR to solve a PDE requiring higher resolution.

RMR is a method of virtualization which exploits the natural behaviors of an analog mesh to enable sharing of the mesh among individual pieces of a decomposed problem. The algorithm first maps the problem to the mesh, providing a coarse-grain solution which is then subdivided into quadrants. Thereafter, the boundary conditions of each quadrant are mapped to the perimeter of the physical mesh, and the process is repeated. Quadrants are continually broken down in the same manner, until the last level of resolution matches the physical mesh resolution requirements. Figure 7 illustrates the spatial resolution of each RMR iteration, followed by quadrant and subquadrant scheduling of the mesh.

When a quadrant's boundary information is mapped to the periphery of the mesh, the computation of a single quadrant

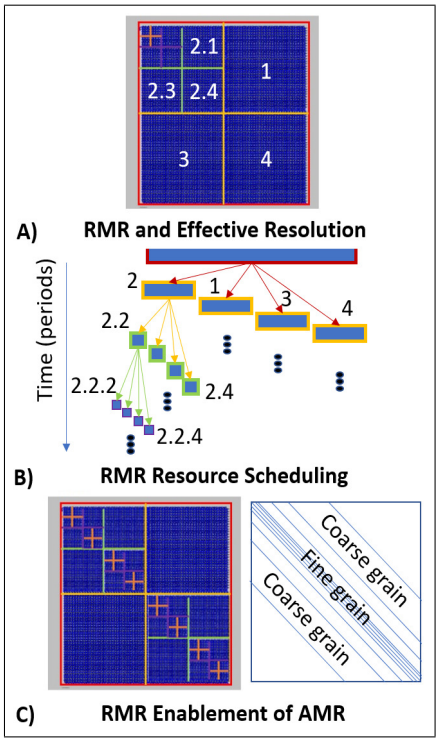

Figure 7: Recursive Mesh Refinement (A), scheduling of mesh resources (B), and its support of AMR (C).

is spread over the entirety of the mesh hardware. This gives the appearance of increased resolution, called *effective resolution*. When the desired resolution is reached, the entirety of the nodes within the mesh are stored. This process is repeated for all quadrants, and results from submeshes stitched together to form a final solution.

There is a natural mismatch in that the number of biases comprising the perimeter of a quadrant is always one half of the biases required to completely set the mesh's perimeter biasing circuitry. Linear interpolation can be accurately used to estimate the missing biases, considering that the missing nodes are, in essence, the center of a resistive divider comprising resistors of $R/2$ resistance [32]. This technique enables the forced matching of perimeter node voltages.

The energy and time required for a RMR calculation are related to the number of calculations required for an effective resolution by Equation 3,

$$n_{calc} = 1 + 4^{n_{levels}}, \tag{3}$$

where n_{levels} is the number of recursive levels needed for a specific effective resolution, calculated in (4).

$$n_{levels} = (resolution_{effective} \div resolution_{mesh}) - 1 \tag{4}$$

Storage requirements for quadrant boundary conditions can be calculated as (5). Registers co-located with the opto-

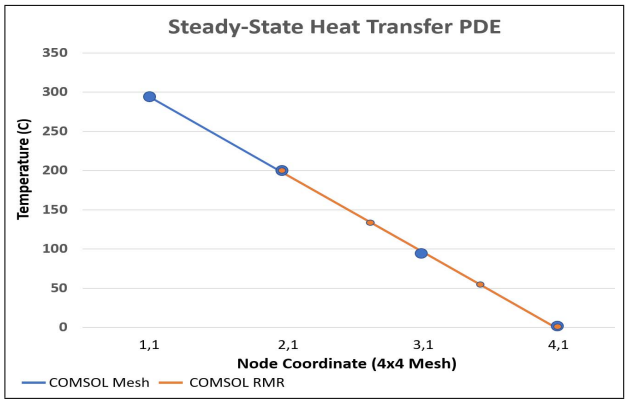

Figure 8: COMSOL RMR results for increasing resolution of a 4x4 mesh.

electronic conversion circuitry provide a storage solution with minimal access time, freeing the system bus. However, any parameter overflow due to large levels of recursion can be stored in main memory through a bulk DMA transfer.

$$memory = 4^{n_{levels}-1} \times diameter_{mesh} \times bits_{perword} \quad (5)$$

Energy, storage and time required for RMR can be reduced using existing techniques for reduced-energy computing, such as AMR. Shown in Figure 7, AMR reduces the energy, time and storage requirements for resolution enhancement operations by limiting the use of RMR to regions of the mesh which require high resolution.

As RMR is required to match a low resolution mesh with a high resolution problem, each iteration of RMR can be viewed as a single pass through an approximate computer. Figure 8 shows our COMSOL [33] simulation results for RMR over a left-side, single-source and right-side, single-sink configuration. Note that the refined submesh results overlay the coarse mesh results, which is indicative of an error-free process. Nonetheless, common error sources inherent in analog mesh computers can reduce accuracy of the final computation. These error sources include: 1) offset due to resolution of the physical mesh, 2) mesh component nonlinearities, and 3) nonlinearities in the biasing and readout circuitry. These error sources can be reduced to primarily resolution-based offset when considering the averaging of noise sources through scaling [4], [7]. Consequently, regarding a single instance of RMR as an error function, X, allows multiple instances of RMR to be viewed as a polynomial function of X. This results in an uncertainty, δR, described by Equation 6 [34]:

$$\delta R = |n| \times \frac{\delta X}{|X|} \times |R|, \quad (6)$$

where R is the solution, and n is the number of recursions through the analog mesh.

2) Linear Interpolation: Interpolation is a fundamental method for estimation of unknown values which reside within a set of known values, requiring only a subset of data points and an equation that describes the system. The well-known linear relationship between voltage and current supports the estimation of intermediate node values using linear interpolation for an electric mesh. ROEs exhibit the same lumped-element behavior as resistors for wavelengths much less than device feature size [7], [23]. This enables the extension of linear interpolation to an optical mesh based on diffusion currents with minimal error introduction. For interpolation, we used a 2D linear spline interpolation [35] provided by the SciPy [36] library.

Resolution matching via linear interpolation requires a single computation by the analog mesh, followed by a series of linear calculations. After reading out the solution for the entirety of the mesh, *grid equivalence* is calculated by dividing the required resolution by the mesh resolution. This is the number of interpolation points that must be calculated between adjacent grid points in the analog mesh, and is related to the number of linear calculations, n_{calc}, by Equation 7.

$$n_{calc} = (X_{mesh} - 1) \times (Y_{mesh} - 1) \times equiv_{grid} \quad (7)$$

As opposed to standard numerical methods, which are limited in their parallelizability, linear interpolation in support of increased resolution is inherently parallelizable. This is due to the availability of a low-resolution mesh solution, which creates a set of boundary points which can easily be expanded in parallel.

3) Hybrid Techniques: Heterogeneous systems, such as microprocessor-based systems with loosely-coupled accelerators, provide adequate resources to support virtualization using a combination of RMR and linear interpolation. Dual-track Virtual Configuration Management (VCM) automates load-balancing between a microprocessor and FPGA accelerator by trading-off resource availability and time-to-solution [37], [13]. Shown in Figure 9, this technique can be extended to analog-mesh virtualization by expanding attributes to include availability and time-to-solution.

IV. Results

A. Evaluation Methodology

Mesh virtualization accuracy for ROC meshes was evaluated using two test cases which represent the spectrum of use cases for computational mesh workloads. Case 1 demonstrates a larger optical mesh than computational mesh, where the optical mesh must compute the solution to simultaneous PDEs. Case 2 demonstrates a smaller optical mesh than computational mesh; for our test purposes, a 4x4 optical mesh must emulate a 64x64 mesh.

Due to the analogous behavior of electrical meshes and metamaterial-based optical meshes [20], [21], mesh simulations for accuracy were executed with SPICE. For all simula-

Table I: Error associated with scaling of virtualized mesh partitions for different PDE boundary conditions.

	PDE	Left to Right			Diagonal			Center Source		
	Scale Dimension	2x2	4x4	8x8	2x2	4x4	8x8	2x2	4x4	8x8
	Free Border Nodes (%)	18.75			35.9			37.5		
	Free Non-Border Nodes (%)	31.25			25			18.75		
Border Nodes	Max Error (%)	0	0	0	0.024	0.02	0.024	0.045	0.017	0.333
	Avg Error (%)	0	0	0	0.007	0.006	0.009	0.025	0.017	0.098
Non-Border Nodes	Max Error (%)	0	0	0	0	0	0.02	0.079	0.076	0.12
	Avg Error (%)	0	0	0	0	0	0.003	0.042	0.034	0.108
Overall Mesh	Max Error (%)	0	0	0	0.024	0.02	0.024	0.079	0.076	0.333
	Avg Error (%)	0	0	0	0.004	0.004	0.005	0.029	0.022	0.101

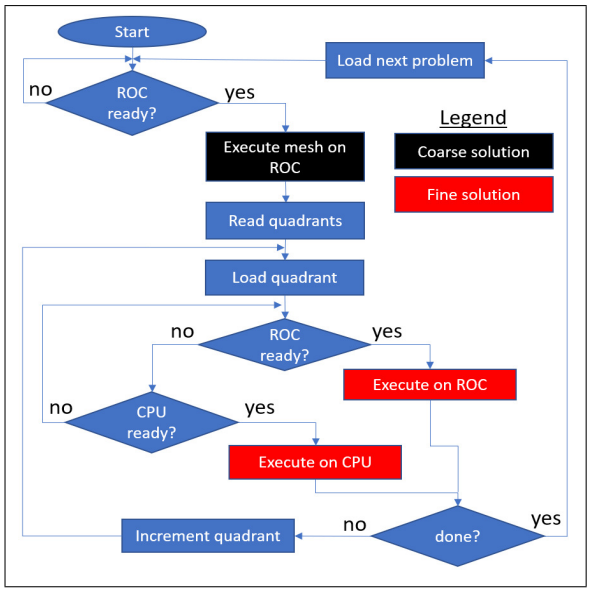

Figure 9: VCM for single round of time share.

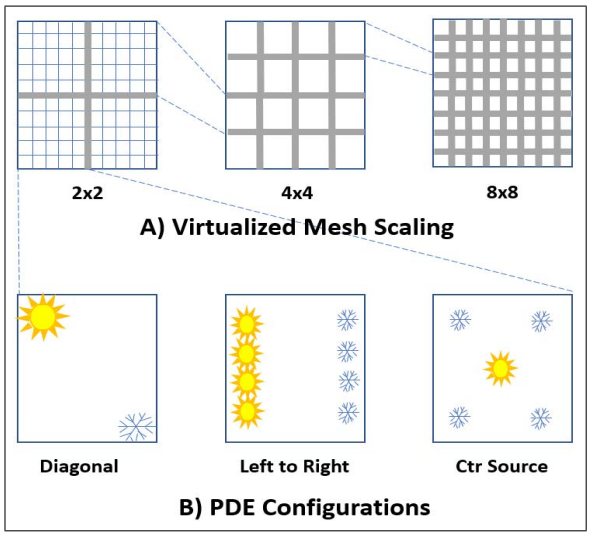

Figure 10: Scalability of virtualized mesh partitions over a single physical mesh (A) and various PDE boundary conditions under consideration (B).

tions, virtualized mesh results were generated and compared with SPICE results for an equivalently-sized physical mesh, which was regarded as the ground truth. Timing modeling of ROC meshes assumes biases that are written in a serial chain running at 100 MHz, and optical mesh settling times were extrapolated from Interconnect [38] simulations of small optical meshes. A detailed analysis about optical network propagation and stabilization delay was done by Anderson *et al.* in [7].

B. Case 1: Computational Mesh Smaller Than Optical Mesh Resolution

Case 1, illustrated in Figure 10, shows the analog mesh computer executing a simultaneous computation of multiple PDEs. Three PDE configurations: **1)** single-source, single-sink with a diagonal diffusion, **2)** left to right lateral diffusion, and **3)** center source were simulated. These configurations were chosen due to the expected variations of current nearest the virtualization boundary.

To study the scalability of our virtualization solution, we divided an optical mesh into 2x2, 4x4 and 8x8 grids of virtual 4x4 submeshes, each with an identical PDE configuration. Virtualization error was calculated by comparing the simulation results of virtualized mesh partitions with that of an isolated mesh of equal size.

Results in Table I show less than 1% error introduced to the computation as a result of virtualization, and a maximum of 0.026% error introduced to the computation. As expected, there is little bilateral influence seen between virtualized mesh partitions due to the high attenuation of the virtualized mesh boundary. Bilateral influence can be seen to scale with the number of free border nodes adjacent to the virtualized mesh boundary. However, this can be reduced by decreasing the distance between stiff nodes with boundary conditions at the virtualized mesh partition border. This effect is best seen when comparing the lateral diffusion PDE with the diagonal diffusion PDE, as the former has a shorter distance between stiff boundary nodes.

Hardware overhead incurred by the setup of the virtualization boundary includes two rows of vertical attenuators and a row of horizontal attenuators. This results in an approximately 20% hardware overhead for the configurations shown. However, the time required for ROC computation of

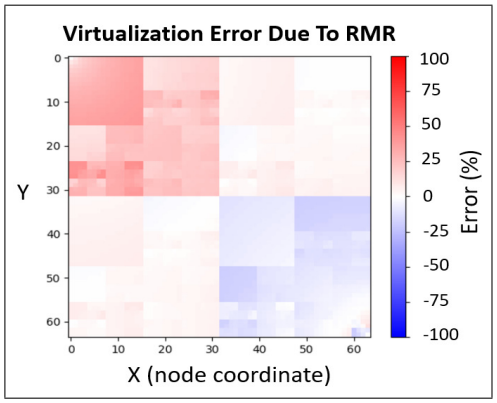

Figure 11: Virtualization error associated with 4x4 emulation of 64x64 using RMR.

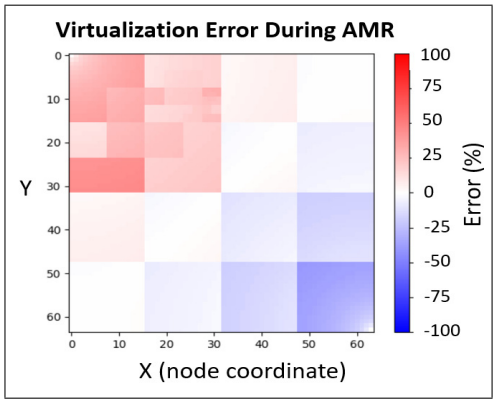

Figure 12: Virtualization error for a 4x4 mesh emulating a 64x64 mesh using RMR with AMR.

all PDEs is 10ms, which is identical to any computation by an 8x8 optical mesh. This is possible because virtualization boundary setup and teardown are hidden as part of the mesh surface configuration.

C. Case 2: Computational Mesh Larger Than Optical Mesh Resolution

The Case 2 simulation environment, shown as the diagonal diffusion PDE shown in Figure 10, addresses a 4x4 mesh which must emulate a 64x64 mesh. These emulations were implemented using RMR with the ROC software stack [39]. Additionally, AMR is represented in this case, due to its common use in numerical solutions.

Figure 11 shows the virtualization error over the entire mesh. Note that virtualization error is highest near the source and sink, and minimizes as regions become more isolated. The nonuniformity of RMR error suggests that standard error correction schemes cannot be used. The largest percentage of nodes were characterized by error less than 20%, with a maximum error at approximately 40% and a minimum error of 0%. The mean error introduced by RMR is 11%, while the median is 8%.

Figure 12 shows the virtualization error caused by combining RMR with AMR. As the single-source, single-sink diagonal configuration showed the most nonlinearity near the sink and source (Figure 4), these regions are characterized as having a high resolution requirement. This is in contrast with regions farther from the source and sink, where regions of lower nonlinearity make a low-resolution mesh more acceptable. Due to the use of AMR, the percentage of error-prone nodes decreases, when compared to full-resolution RMR, while the minimum and maximum errors stay constant. The mean error introduced by AMR is 12%, while the median is 6%.

Table II compares the time-to-completion of 3 implementations of PDE solvers: 1) a 4x4 ROC RMR emulation of a larger mesh, 2) a single ROC mesh of comparable size and

Table II: Time-to-completion of 4x4 RMR, a single ROC mesh of comparable size [7], and execution by a CPU.

size	8x8	16x16	32x32	64x64
ROC (ms)	0.0151	0.0200	0.0350	0.0500
RMR (ms)	0.058139	0.314	1.913	12.896
CPU[1] (ms)	3000	4000	5000	5000

3) a CPU implementation using COMSOL on an Intel Xeon E5603 running at 1.6 GHz. Time-to-completion for ROC includes time required for ROC configuration, execution and readout. Time-to-completion for the CPU implementation is as reported by COMSOL.

Each doubling of the mesh dimension results in an increased time-to-solution for the RMR-based mesh calculation. This is to be expected, as the overhead of mesh virtualization follows Equation 3. However, despite this overhead, virtualization of the ROC mesh performs faster than sequential software by two orders of magnitude.

The overhead associated with RMR results in time-to-completion higher than those of a physical mesh of comparable size. This is in contrast to mesh complexity, where RMR allows ROC to stay constant, while physically scaling ROC results in a complexity of $O(n^2)$. Memory complexity due to RMR boundary storage requirements is $O(n)$. Analysis of RMR and sequential execution of Laplace's equation showed time complexities of $O(log^2(n) \times t)$ and $O(n^2 \times m)$, respectively, where t includes the per-run cost of mesh setup, execution and readout, and m is the number of iterations.

V. Conclusion

While new post-Moore's Law computer architectures are being introduced, virtualization environments need to be developed for these innovative architectures to enjoy mainstream success. Here, we have introduced architectural support for virtualization of an analog mesh computer and applied it to the reconfigurable optical computer, or ROC. For

[1]Time-to-Completion as reported by COMSOL.

a high-resolution mesh, our virtualization technique enables both space-sharing and time-sharing of mesh resources. Additionally, we proposed recursive mesh refinement, or RMR, to support a fixed-size mesh's emulation of larger meshes.

We simulated multiple use cases intended to show the scaling of ROC to match the resolution requirements of PDEs of various sizes. For small PDEs, ROC was able to support simultaneous computations with less than 1% error. The large-PDE use case illustrated RMR's ability to support high-resolution PDE requirements with an 8% median error. We also applied RMR in support of adaptive mesh refinement, which introduced a median 6% error to the PDE computation.

Timing simulations compared various ROC meshes with corresponding virtualized meshes and microprocessor execution. When compared to CPU execution, ROC virtualization showed a reduction in time-to-solution by two orders of magnitude.

ACKNOWLEDGMENT

This work is funded by the NSF RAISE program as award number 1748294 under the NSF EPMD-ElectroPhotonic Mag Devices, CSR-Computer Systems Research, Networking Technology and Systems. Additionally, the authors would like to thank the anonymous reviewers for their insightful comments.

REFERENCES

[1] E. Strohmaier, J. Dongarra, H. Simon, and M. Meuer. Top ranked system list. [Online]. Available: https://www.top500.org/lists/2019/11/

[2] J. Dongarra. Current trends in high performance computing and challenges for the future. [Online]. Available: https://www.acm.org/binaries/content/assets/education/lc-monthly-bulletins/january2017.html

[3] I. Richter, K. Pas, X. Guo, R. Patel, J. Liu, E. Ipek, , and E. G. Friedman, "Memristive accelerator for extreme scale linear solvers," in *Proceedings of the Government Microcircuit Applications and Critical Technology Conference*, ser. GOMACTECH '15, 2015.

[4] G. Liebmann, "Solution of partial differential equations with a resistance network analogue," *BRITISH JOURNAL OF APPLIED PHYSICS*, 1950.

[5] J. Ramirez-Angulo and M. R. DeYong, "Digitally-configurable analog vlsi chip and method for real-time solution of partial differential equations," oct 2000, patent No. US6141676, Filed July 22, 1998, Issued October 31, 2000.

[6] T. El-Ghazawi, V. J. Sorger, S. Sun, A.-H. A. Badawy, and V. K. Narayana, "Reconfigurable optical computer," jun 2019, patent No. US10318680B2, Filed December 5th., 2017, Issued June. 8th., 2019.

[7] J. Anderson, E. Kayraklioglu, S. Sun, J. Crandall, Y. Alkabani, V. Narayana, V. Sorger, and T. El-Ghazawi, "Roc: A reconfigurable optical computer for simulating physical processes," *ACM Transactions on Parallel Computing*, vol. 7, 2020.

[8] G. J. Popek and R. P. Goldberg, "Formal requirements for virtualizable third generation architectures," *Communications of the ACM*, vol. 17, no. 7, July 1974.

[9] A. Whitaker, M. Shaw, and S. Gribble, "Denali: Lightweight virtual machines for distributed and networked applications," 03 2002.

[10] A. Vaishnav, K. D. Pham, and D. Koch, "A survey on fpga virtualization," in *Proceedings of the 2018 28th International Conference on Field Programmable Logic and Applications*, ser. FPL '18. IEEE, 2018.

[11] M. Kotsifakou, P. Srivastava, M. D. Sinclair, R. Komuravelli, V. Adve, and S. Adve, "Hpvm: heterogeneous parallel virtual machine," in *Proceedings of the 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, ser. PPoPP '18, 2018.

[12] E. El-Araby, V. K. Narayana, and T. El-Ghazawi, "Space and time sharing of reconfigurable hardware for accelerated parallel processing," *Reconfigurable Computing: Architectures, Tools and Applications. ARC 2010. Lecture Notes in Computer Science*, vol. 5992, 2010.

[13] M. Taher and T. El-Ghazawi, "Virtual configuration management: A technique for partial runtime reconfiguration," *IEEE Transactions on Computers*, vol. 58, no. 10, October 2009.

[14] H. Lee and W. Schiesser, *Ordinary and Partial Differential Equation Routines in C, C++, Fortran, Java, Maple and MATLAB*. Boca Raton, FL, USA: CRC Press, 2003.

[15] Y. Zhao, "Lattice boltzmann based pde solver on the gpu," *The Visual Computer*, vol. 24, 2008.

[16] L. Pinuel, I. Martin, and F. Tirado, "A special-purpose parallel computer for solving partial differential equations," in *Proceedings of the Sixth Euromicro Workshop on Parallel and Distributed Processing*, ser. PDP '98. IEEE, 1998.

[17] M. J. Berger and J. Oliger, "Adaptive mesh refinement for hyperbolic partial differential equations," *Journal of Computational Physics*, vol. 53, no. 3, pp. 484–512, 1984.

[18] P. Palmer, A. Copson, and S. Redshaw, "Investigations into the use of an electrical resistance analogue for the solution of certain oscillatory-flow problems," AERONAUTICAL RESEARCH COUNCIL, Reports and Memoranda 312, 1959.

[19] H. Yin, *Application of Resistivity-Tool-Response Modeling For Formation Evaluation: AAPG Archie Series*. American Association of Petroleum Geologists, 2011, vol. 2, no. 2.

[20] N. Engheta, A. Salandrino, and A. Alù, "Circuit elements at optical frequencies: Nanoinductors, nanocapacitors, and nanoresistors," *Phys. Rev. Lett.*, vol. 95, p. 095504, Aug 2005. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.95.095504

[21] Y. Gui, M. Miscuglio, Z. Ma, M. H. Tahersima, S. Sun, R. Amin, H. Dalir, and V. J. Sorger, "Towards integrated metatronics: a holistic approach on precise optical and electrical properties of indium tin oxide," *Scientific Reports*, vol. 11279, no. 9, August 2019.

[22] Z. Ma, Z. Li, K. Liu, C. Ye, and V. J. Sorger, "Indium-tin-oxide for high-performance electro-optic modulation," *Nanophotonics*, no. 4, March 2015.

[23] N. Engheta, "Circuits with light at nanoscales: Optical nanocircuits inspired by metamaterials," *Science*, vol. 317, no. 5845, pp. 1698–1702, 2007. [Online]. Available: https://science.sciencemag.org/content/317/5845/1698

[24] C. Reano and F. Silla, "A comparative performance analysis of remote gpu virtualization over three generations of gpus," in *Proceedings of the 2017 46th International Conference on Parallel Processing Workshops*, ser. ICPPW '17. IEEE, 2017.

[25] L. Liu, J. Zhu, Z. Li, Y. Lu, Y. Deng, J. Han, S. Yin, and S. Wei, "A survey of coarse-grained reconfigurable architecture and design: Taxonomy, challenges, and applications," *ACM Computing Surveys*, vol. 52, no. 6, October 2019.

[26] H. Kang, M. Le, and S. Tao, "Container and microservice driven design for cloud infrastructure devops," in *Proceedings of the 2016 IEEE International Conference on Cloud Engineering*, ser. IC2E 2016. IEEE, 2016.

[27] S. A. Fahmy, K. Vipin, and S. Shreejith, "Virtualized fpga accelerators for efficient cloud computing," in *Proceedings of the 2015 IEEE 7th International Conference on Cloud Computing Technology and Science*, ser. ICCTS '15, vol. 1. IEEE, 2015.

[28] C. Siemers, "Reconfigurable computing between classifications and metrics - the approach of space/time-scheduling," in *Proceedings of the The Roadmap to Reconfigurable Computing, 10th International Workshop on Field-Programmable Logic and Applications*, ser. FPL '00, vol. 1. ACM, 2000.

[29] J. Rabaey. The spice page. [Online]. Available: http://bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/f

[30] V. J. Sorger, N. D. Lanzillotti-Kimura, R.-M. Ma, and X. Zhang, "Ultra-compact silicon nanophotonic modulator with broadband response," *Nanophotonics*, no. 1, May 2012.

[31] T. El-Ghazawi, P. Chalermwat, and J. L. Moigne, "Wavelet-based image registration on parallel computers," in *Proceedings of the 1997 ACM/IEEE conference on Supercomputing*, ser. SC '97, 1997.

[32] A. Agarwal and J. Lang, "course materials for 6.002 circuits and electronics," 2007, spring 2007. MIT OpenCourseWare. [Online]. Available: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-002-circuits-and-electronics-spring-2007/video-lectures/6002˙l1.pdf

[33] P.-C. Ung. Comsol blog: Heat transfer in deformed solids. [Online]. Available: https://www.comsol.com/blogs/heat-transfer-deformed-solids/

[34] W. Rueckner, R. Hart, Y. Synkova, and J. Peidle, "Summary of rules for error propagation," 2019, harvard Instructional Physics Lab. [Online]. Available: https://sites.fas.harvard.edu/ sc-phys/nsta/error˙propagation.pdf

[35] I. Schoenberg, *Cardinal Spline Interpolation*, ser. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1973.

[36] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors, "SciPy 1.0–Fundamental Algorithms for Scientific Computing in Python," *arXiv e-prints*, p. arXiv:1907.10121, Jul 2019.

[37] T. El-Ghazawi, "Virtual configuration management for efficient use of reconfigurable hardware," jul 2009, patent No. US20090187733A1, Filed March 3rd., 2007, Issued July. 23rd., 2009.

[38] R. Goldman. Lumerical 2020a release speeds photonic design through high performance computing. [Online]. Available: https://www.prweb.com/releases/prweb16705671.htm

[39] E. Kayraklioglu, J. Anderson, H. Reza-Imani, V. Sorger, and T. El-Ghazawi, "Software stack for an analog mesh computer: The case of a nanophotonic pde accelerator," in *Proceedings of the 2020 International Conference on Computing Frontiers*, ser. CF '20. ACM, 2020.