

Stitching FPGA Dies into a Unified Pipeline for Seismic Stencil Acceleration

Lucas R. Cilento¹, Edson S. Gomi¹

¹Escola Politécnica – Universidade de São Paulo (USP)
São Paulo – SP – Brazil

cilento@usp.br, gomi@usp.br

Abstract. *Finite-difference solvers for the 3D acoustic wave equation are widely used in seismic imaging. We present a multi-die FPGA architecture that exploits the three dies of a Xilinx Alveo U55C as a single computational pipeline interconnected by on-chip AXI-Stream FIFOs. By streaming each tile across all dies in one pass, the design eliminates intermediate off-chip memory transfers and increases the effective stencil depth available to the computation. Experiments show approximately $1.8\times$ less data movement than a conventional replicated design, increasing sustained performance from 96 to 175 GFLOP/s.*

1. Introduction

Seismic imaging methods such as Full Waveform Inversion (FWI) are computationally intensive workloads that solve the acoustic wave equation on large 3D grids using explicit finite-difference stencils [Virieux and Operto 2009]. Their regular memory-access pattern makes them well-suited to deeply pipelined implementations on Field-Programmable Gate Arrays (FPGAs). However, multi-die FPGAs introduce additional constraints. In the Xilinx Alveo U55C, three dies are connected by limited inter-die links, and each kernel is restricted to a single die [AMD/Xilinx 2022]. Consequently, a kernel can access only a subset of the available resources. The conventional approach uses the same kernel across all dies and processes tiles independently, avoiding inter-die communication.

In this work, we instead explore chaining the three dies as stages of a unified pipeline: wavefield data are streamed across die boundaries, so each tile is processed sequentially by all dies. We evaluate when this approach outperforms replication.

Previous FPGA stencil accelerators have focused on single-die devices [Zohouri et al. 2018], multi-FPGA systems [Waidyasooriya and Hariyama 2019, Del Sozzo et al. 2024], or alternative wave-simulation methods [Gourounas et al. 2023]. To our knowledge, using multiple dies of a single FPGA as a unified streaming pipeline for finite-difference seismic imaging has not been previously investigated.

2. Background

Finite-difference stencil: The acoustic wave equation is discretized with a second-order explicit scheme, yielding a 7-point stencil of radius $r = 1$: at each time step, the pressure at a grid point depends on its current value, its previous value, and its six axis neighbors:

$$p_{i,j,k}^{n+1} = \Delta t^2 c_{i,j,k}^2 \nabla_h^2 p_{i,j,k}^n + 2p_{i,j,k}^n - p_{i,j,k}^{n-1}, \quad (1)$$

where ∇_h^2 is the discrete Laplacian. Since every grid point follows the same access pattern, all dependencies are known in advance, making it well suited to FPGA pipelines.

Streaming stencil pipelines: We use the standard FPGA streaming approach [Waidyasooriya and Hariyama 2019]. Grid cells enter the device as a continuous stream, while pipelined stages compute successive time steps; several lanes process cells in parallel. Neighboring values come from a shift-register structure that moves with the stream, with storage proportional to $2N_X N_Y$, independent of the streamed dimension (Z).

Overlapped tiling: Practical seismic grids are much larger than the FPGA’s on-chip memory, so the domain is divided into tiles stored off-chip between passes. When a pipeline advances multiple time steps in one pass, cells near the tile boundaries become inaccurate: some of their required neighbors lie outside the tile. To preserve correctness, each tile is extended with a halo whose width equals the number of time steps per pass, since the stencil propagates information r cells per step. The FPGA processes the padded tile, but only the central region is written back; the contaminated halo is discarded and, since adjacent tiles overlap by exactly the halo width, recovered from their interiors.

3. Architecture

The kernel exposes two parameters: temporal depth (D) and spatial width (K). Depth is the number of pipelined stages, mainly consuming on-chip Block RAM (BRAM), while width is the number of cells processed per cycle, mainly consuming Digital Signal Processing (DSP) blocks. Drawing on different resources, they cannot be traded directly.

Increasing D enlarges the halo region, reducing the useful tile area and increasing redundant computations. K has no such penalty and is limited only by bandwidth.

When the domain fits in the card’s High Bandwidth Memory (HBM), bandwidth supports wide kernels, allowing K to approach the DSP limit. In this scenario, three kernels can be replicated across the three dies to process different tiles concurrently. However, production seismic domains exceed HBM capacity, requiring tile transfers over PCI Express (PCIe), which becomes the bottleneck and sustains about one full-width stream.

To address this limitation, the three dies are connected as a single pipeline rather than used as independent replicas. Only the first kernel receives data from the CPU, while subsequent kernels consume data from the previous stage through on-chip streams; a single input stream thus feeds all $3D$ pipeline stages, yielding an effective depth of $D_{\text{eff}} = 3D$. Replication cannot provide the same benefit because each replica requires its own external stream. Extending the halo from D to $3D$ provides sufficient boundary information for continuous processing across dies, while inter-die communication is limited to shallow first-in-first-out (FIFO) buffers, eliminating intermediate HBM storage.

The trade-off is a larger halo, which reduces the useful fraction of each tile:

$$\eta = \frac{(N_X - 2D_{\text{eff}})(N_Y - 2D_{\text{eff}})}{N_X N_Y}. \quad (2)$$

Section 4 shows this reduction is outweighed by deeper pipelines and fewer passes.

The final design uses the maximum stream width sustained by PCIe, $K = 4$, and increases the depth until each die’s BRAM is nearly exhausted at $D = 8$, yielding

$D_{\text{eff}} = 24$ across the three dies. Although BRAM would permit $N \approx 220$, we select $N = 160$ so three tiles fit in HBM together, overlapping PCIe transfers with computation.

The hardware implementation consists of three kernels that share the same stencil body but have different interfaces (Figure 1). `stmi3d_first` reads data from HBM and emits an Advanced eXtensible Interface (AXI) stream, `stmi3d_chain` forwards data between dies, and `stmi3d_last` writes the final result back to HBM. One kernel sits on each die, adjacent kernels joined by shallow FIFOs. The CPU still prepares a single tile, which the FPGA executes as one pipeline spanning the three dies.

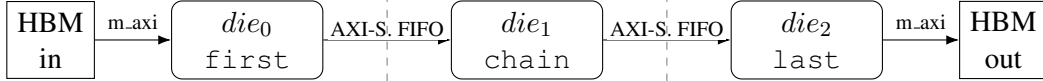


Figure 1. One kernel per die; AXI-Stream FIFOs cross dashed die boundaries.

4. Evaluation

We evaluate the chained pipeline on an Alveo U55C at 250 MHz. Over a full 2048^3 simulation, the output matches a single-precision reference solution, and symmetric inputs preserve symmetry, confirming correctness.

For each tile, the CPU assembles and transfers data in about 70 ms (20 ms assembly + 50 ms PCIe write), while the FPGA processes it in 65 ms. PCIe read-back overlaps with the next transfer, and CPU preparation overlaps with kernel execution. The critical path is therefore the CPU, resulting in approximately 93% kernel utilization.

The useful work per tile is proportional to $D_{\text{eff}}\eta$. The chained design achieves $D_{\text{eff}} = 24$ with $\eta \approx 0.49$, whereas replication reaches only $D = 8$ with $\eta \approx 0.81$. Consequently, the data volume required for a fixed simulation scales as

$$\frac{V_{\text{pipe}}}{V_{\text{repl}}} = \frac{D_{\text{repl}} \eta_{\text{repl}}}{D_{\text{eff}} \eta_{\text{pipe}}} = \frac{8 \times 0.81}{24 \times 0.49} \approx 0.55, \quad (3)$$

meaning the chained pipeline transfers about $1.8\times$ less data than replication. Since execution is CPU-bound, this reduction directly improves throughput.

At $K = 4$ and $D_{\text{eff}} = 24$, the design evaluates 96 stencils per cycle, corresponding to a peak performance of 384 GFLOP/s, or 24 Gcells/s, at 250 MHz. Accounting for the valid fraction ($\eta \approx 0.49$) and 93% kernel utilization, the sustained useful throughput reaches approximately 175 GFLOP/s (10.9 Gcells/s of grid-point updates). Replication, which transfers $1.8\times$ more data, sustains about 96 GFLOP/s (6.0 Gcells/s). These rates are in the range of published FPGA stencil accelerators [Zohouri et al. 2018, Waidyasooriya and Hariyama 2019] and GPU finite-difference kernels [Gourounas et al. 2023], though published results target higher-order stencils.

Table 1 summarizes the comparison and reports the per-die usage of lookup tables (LUT), flip-flops (FF), BRAM, UltraRAM (URAM), and DSP blocks: both designs share the same three kernels and differ only at their interfaces, so footprints are nearly identical, and each die is bounded by routing congestion of its $D \times K = 32$ floating-point datapaths rather than by resources; the pipeline’s gain comes from data movement, not area.

Table 1. Pipeline vs. replication (2048³ domain, 250 MHz) and per-die resources.

	Pipeline	Replication	LUT	100k
Effective depth D_{eff}	24	8	FF	164k
Valid fraction η	0.49	0.81	BRAM	≈ 400
Relative data moved	1.0	1.8	URAM	0
Throughput (GFLOP/s)	175	96	DSP	1,216
Useful rate (Gcells/s)	10.9	6.0		

5. Conclusion

We chained the three dies of the multi-die FPGA into a single AXI-Stream pipeline instead of using them as independent replicas. Matching the halo depth to the pipeline depth hides die boundaries, requiring only shallow FIFOs and no off-chip buffering. The resulting pipeline is three times deeper than a single replica and, for large CPU-bound domains, moves about $1.8\times$ less data per simulation, increasing sustained performance from ≈ 96 to ≈ 175 GFLOP/s. Replication remains advantageous only when the domain fits entirely in HBM. Production seismic codes typically employ high-order stencils ($r \geq 4$), which widen the halo proportionally and reduce the feasible pipeline depth, shifting the balance between chaining and replication; supporting them, along with hybrid replication–chaining schemes and energy measurement, is left as future work.

Acknowledgments: This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001. FPGA resources were provided by FAPESP Thematic Project 19/26702-8.

References

- AMD/Xilinx (2022). *UltraFast Design Methodology Guide for FPGAs and SoCs (UG949)*. AMD.
- Del Sozzo, E., Conficconi, D., and Sano, K. (2024). Across time and space: Senju’s approach for scaling iterative stencil loop accelerators on single and multiple FPGAs. *ACM Trans. on Reconfigurable Technology and Systems*, 17(2):1–33.
- Gourounas, D., Hanindhito, B., Fathi, A., Trenev, D., John, L. K., and Gerstlauer, A. (2023). LAWS: Large-scale accelerated wave simulations on FPGAs. In *Proc. ACM/SIGDA Int. Symp. on Field-Programmable Gate Arrays (FPGA)*.
- Virieux, J. and Operto, S. (2009). An overview of full-waveform inversion in exploration geophysics. *Geophysics*, 74(6):WCC1–WCC26.
- Waidyasooriya, H. M. and Hariyama, M. (2019). Multi-FPGA accelerator architecture for stencil computation exploiting spatial and temporal scalability. *IEEE Access*, 7:53188–53201.
- Zohouri, H. R., Podobas, A., and Matsuoka, S. (2018). Combined spatial and temporal blocking for high-performance stencil computation on FPGAs using OpenCL. In *Proc. ACM/SIGDA Int. Symp. on Field-Programmable Gate Arrays (FPGA)*, pages 153–162.