Sistema FIEB | SENAI CIMATEC

PELO FUTURO DA INOVAÇÃO

Xilinx 3D RTM on FPGA

Accelerating 3D Seismic Modeling with Alveo U280 FPGA

SENAI CIMATEC Supercomputing Center for Industrial Innovation
October 2020
Salvador, BA

SENAI CIMATEC

Supercomputing Center

CIMATEC 2 - 3rd floor

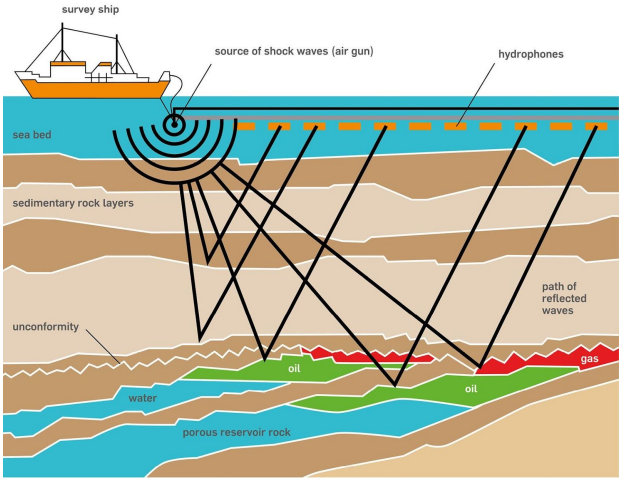
Av. Orlando Gomes, 1845 - Piata 41650-010 Salvador - Bahia - Brasil

Phones: (71) 3462-9587 / (71) 3462-9583

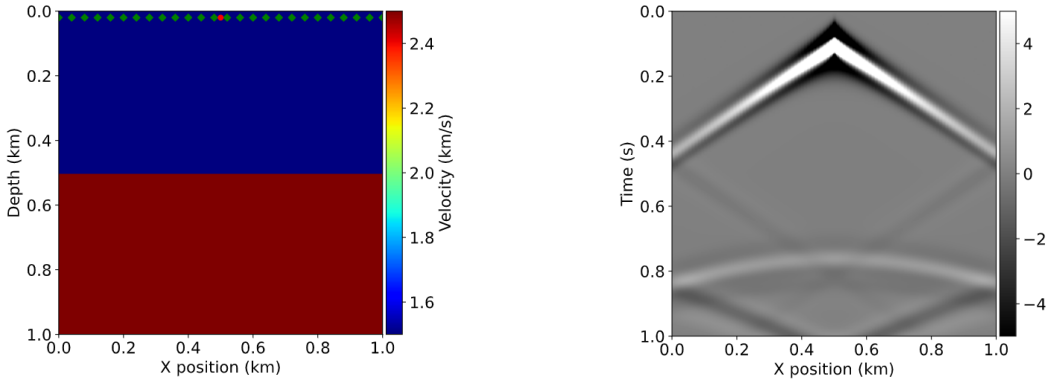
Contents

1. Synthetic Modeling Relevance in Seismic Survey Workflow

A Seismic Survey is a process widely used by the Oil&Gas Industry for investigating subterranean structures to identify underground oil reserves and determine optimal drilling points for minimizing exploration costs. The survey process is divided into two different phases: data-collection and post-processing. During the data-collection phase, a series of controlled explosions (i.e. *seismic shot*) generate pressure waves that travel into the analyzed domain; Simultaneously, an array of seismic sensors measures the reflected waves, and the time elapsed between its injection and arrival. Figure 1a illustrates the data-collection process for a typical offshore survey.



a)



b) c)

Figure 1: a) Offshore seismic data-collection illustration; b) 2-layers bidimensional velocity model; c) Synthetic Shot Record example.

A collection of seismic measurements is associated with each shot and stored on a Shot Record data structure. Figure 1c shows an example of a synthetic 2D Shot Record for a simple 2-layers underground structure (Fig. 1b). In this example, a seismic shot was placed in the top-center of the analyzed domain; the reflected waves were collected during the first second after the shot.

During the post-processing phase, Shot Records are filtered and later used as input for imaging algorithms such as Reverse Time Migration (RTM). Besides Shot Records, RTM algorithms also require discrete representations of the shot impulse (Ricker Wavelets) and a grid containing acoustic velocities inside the analyzed volume (Velocity Model). Its goal is to enhance the contrast between velocity layers by cross-correlating reconstructed Shot Record waves with the simulated impulse waves across the domain grid. As a result, it produces an output grid image with higher resolution than the one used as velocity input, allowing researchers to determine the composition of geological layers with more precision.

Designing and tuning RTM algorithms require evaluating its output with different input sets, varying velocity grid, and shot record dimensions and complexities. As real-world records are often large and difficult to obtain, RTM algorithms are designed and validated using synthetic seismic data on lab environments and only later put into the production workflow. Therefore, the ability to simulate acoustic waves propagation across a velocity grid and generate artificial synthetic records with different length, precision, and geometry is essential for RTM prototyping.

The process of generating synthetic shot records is called Seismic Modeling. Seismic Modeling may require high computing power and large quantities of storage resources depending on the analyzed volume sizes. It can take hours to execute in pure CPU implementations, increasing energy demand even in well-tunned HPC clusters.

2. 3D Acoustic Seismic Modeling

Reverse Time Migration (RTM) is quickly becoming the standard for high-end imaging. At the core of the RTM algorithm is a seismic modeling kernel which simulates the propagation of an acoustic wave across a 3D seismic velocities grid. This modeling kernel can also be used for generating synthetic shot records with any arbitrary acquisition geometry. In this process, the user specifies a mesh of receiver objects that are placed inside the analyzed volume at a certain depth Rz. The kernel than propagates the source wavefield from t=0 to $P\,[t+1] = 2P$ t= tmax, and pressure values are stored for every t at all receiver points, thus emulating the operation of seismic sensors (e.g. hydrophones, geophones, etc.) as in the data-collection phase. Figure 2 shows how the acquisition geometry is organized at a 3D volume.

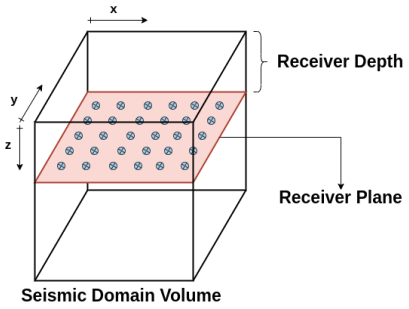


Figure 2: Seismic Receivers acquisition geometry.

The modeling kernel used in this report is a discretized version of the 3D Acoustic Wave Equation for P-waves only, defined by Equation 1. The discretization is implemented using a finite-differences operator for spatial derivatives as shown in Equation 2. A virtual absorbing border is added to the seismic volume and used for attenuating energy and avoiding wave reflections back inside the interest region, thus emulating an infinite domain.

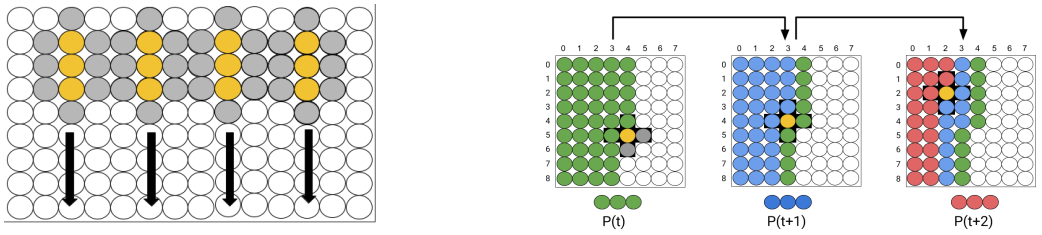
$$\nabla^2 P = \frac{1}{c^2}\frac{\partial^2 P}{\partial t^2} \qquad \text{(Eq. 1)}$$

- P: P(x,y,z,t) is the pressure field;
- c: c(x,y,z) is the seismic velocity;
- ∇^2 : is the Laplacian operator of spatial derivatives;

$$P[\,t+1\,] = 2P[\,t\,] - P[\,t-1\,] + c^2\, L\, \Delta t^2 \qquad \text{(Eq. 2)}$$

- L: is the discrete implementation of the Laplacian operator;
- Δt^2: is the time derivative (time step);

The Seismic Modeling process is a memory-bound problem. The finite-differences stencil operator requires that at least *3*(k+1)* neighbor points are fetched in memory for every pressure point update, thus limiting performance to available bandwidth. In the other hand, it presents two significant opportunities for parallelism: one in the spatial domain, given that all points can be updated simultaneously within the same time iteration; and another in the temporal domain, given that as soon as a pressure point is updated it can be immediately used as input for the next time iteration. These two domains of intrinsic parallelism, illustrated in Figure 3, can be exploited by acceleration kernels to increase propagation throughput.



a) b)

Figure 3: Intrinsic parallelism of the wave propagation modeling kernel for a 2D workload.

3. Accelerating Seismic Modeling with Alveo FPGA

Alveo U280 FPGA boards are equipped with 8GB of HBM2 distributed in 32 memory banks of 256MB, each one with an independent access interface. This architectural feature allowed our Seismic Modeling FPGA kernel to explore spatial parallelism by splitting both pressure and velocity 3D grids across the 32 memory banks and fetching several regions of the seismic volume in parallel. Temporal parallelism, by its turn, is achieved by streaming updated pressure fields into on-chip shift-register memories, allowing multiple timesteps to be computed at the same clock cycle.

Figure 4 illustrates the FPGA kernel internal architecture. The image grid is read sequentially in the *x* and *y* directions. Each Streaming Module (SM) computes up to 16 pressure points on the same time step. The points produced by each SM are shifted into the next SM as input to the next time step. The last cascaded SM_{n-1} stores the grid points back to HBM banks.

One important restriction for this structure is that the total number of time steps must be an integer multiple of the number of SMs (*nFSM* parameter). The number of points updated per time iteration is defined by synthesis parameter *nPEz*, while the number of consecutive x-y planes stored on local memory is defined by *nPEx*. The maximum values for *nPEz*, *nPEx*, and *nFSM* parameters will depend on the available hardware resources on the target FPGA. Table 1 shows resource utilization for a kernel synthesized with ***nPEz=4***, ***nPEx=2*** and ***nFSM=2***. This configuration was used to generate the hardware binaries used in performance comparison against GPU presented in the next section.

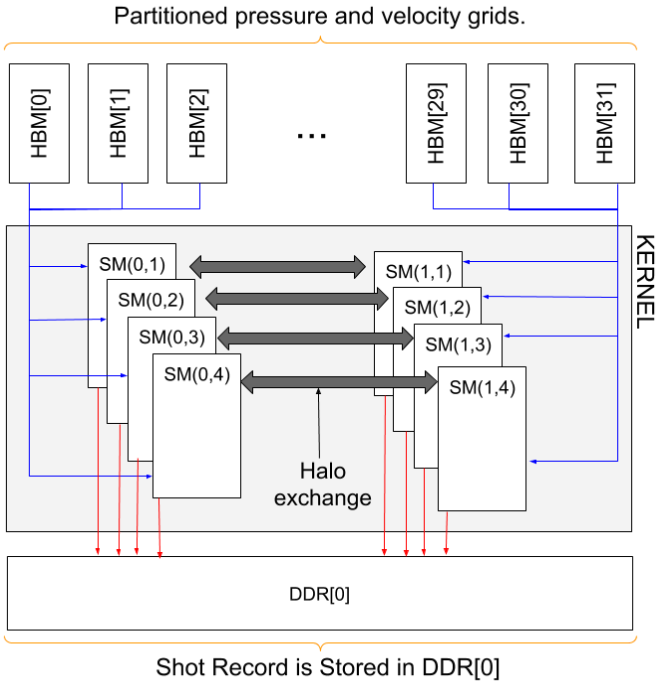


Figure 4: 3D Seismic Modeling FPGA Kernel Architecture.

Table 1: Resource utilization for kernels compiled with *nPEz=4*, *nPEx=2* and *nFSM*=2.

Name	BRAM_18K	DSP	FF	LUT	URAM
DSP	-	-		-	
Expression	-	-	0	3088	
FIFO	-	-	-	-	
Instance	804	2081	309914	182669	200
Memory	0	-	896	48	
Multiplexer	-	-	-	1938	
Register	-	-	7115	352	
Total	804	2081	317925	188095	200
Available on SRL0	1344	3008	869120	434560	320
Utilization on SLR0 (%)	59	69	36	43	62
Available on U280 FPGA	4032	9024	2607360	1303680	960
Utilization (%)	19	23	12	14	20

4. Performance Comparison with NVidia's V100 GPU.

Graphic Processing Units (GPU) are currently the standard acceleration devices in HPC environments, especially for scientific computing. Their massively parallel architecture, combined with intuitive software compiler and API offer great flexibility for thread-based parallelization over larger workload sizes. We have compared our FPGA kernel performance

on U280 against an equivalent implementation for NVidia's V100 GPU, one of the top GPU devices in the market.

Our analysis considered two significant performance aspects for real-world production environments: execution time and energy consumption. The execution time aspect tells how fast the synthetic Shot Record is generated by the kernel, while the energy consumption determines how much electric power the accelerator needs to achieve it. Here we consider only the energy consumed by both acceleration boards via PCIe bus.

Four different 3D velocity models were used as input for FPGA and GPU kernels, varying the workload sizes. Table 2 shows the dimensions of each model. All velocity and pressure values are represented using 32-bits single precision floating point representation.

Table 2: 3D velocity models used for FPGA and GPU benchmarking.

Name	Grid	(NT)	Border	Grid Size (MB)	Shot Record (MB)
3-LAYERS	(64,74,84)	1024	16	3 x 4.720	19.40
SEG-EAGE-R	(214,214,210)	2368	21	3 x 66.06	433.78
OVERTHRUST	(402,242,432)	3224	16	3x 220.71	1254.57
SEG-EAGE	(676,676,210)	3300	21	3 x 519.65	6032.08

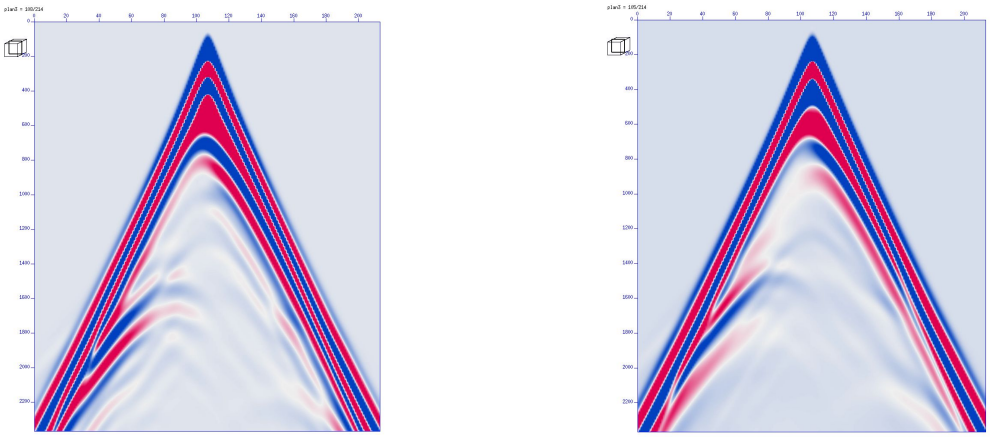
The performance evaluation experiment was performed using the following methodology:

1. A single seismic shot was positioned at the top center coordinates of each model. For instance, for the SEG-EAGE-R model, the shot coordinates were S_{xyz} = (107,107,0);
2. The source wavefield was propagated across the volume grid throughout NT time steps;
3. The receiver objects were placed along the entire top plane of the seismic volume for all models, at depth 0. For instance, the acquisition geometry for OVERTHRUST model included a total of *402x242=97284* receivers;
4. The Shot Record was generated by collecting wavefield pressure values on every time step at all receiver points;
5. We have visually compared the output records for FPGA and GPU kernels against a reference CPU implementation for validation;
6. The execution time is the time elapsed (measured in seconds) between the start of the host program and its end.
7. The same host program was used for controlling FPGA and GPU kernels. The CUDA API was used to control the V100, while Xilinx Runtime OpenCL Library was used for U280;
8. A series of power samples were collected during the program execution using each device's management utility (i.e. *xbutil* for U280 and *nvidia-smi* for V100).
9. An energy consumption estimation, measured in Watt Hour (Wh), was calculated using the collected power samples according to Eq. 3.

$$E = \frac{H_t}{N} \cdot \sum_{k=0}^{N-1} p(\kappa) \qquad \text{(Eq. 3)}$$

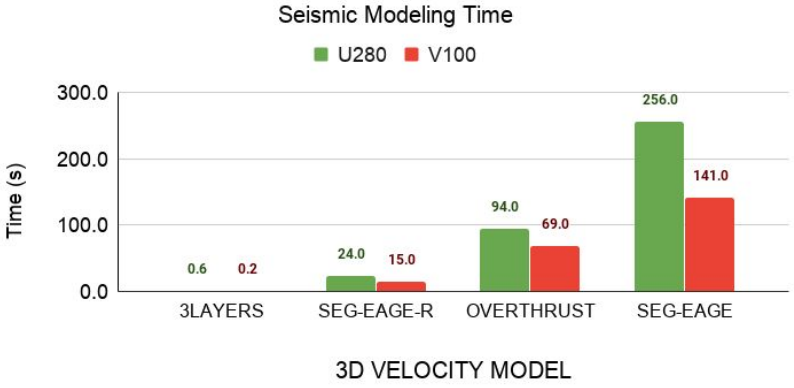
- E: energy consumption measured in Wh;
- H_t: execution time measured in Hours;
- N: number of power samples collected during Seismic Modeling execution;
- $p(k)$: k-th power sample (W);

Figure 5 shows a 2D section of the Shot Record generated for the SEG-EAGE-R model, while Figure 6 shows the execution time and the energy consumption estimation for all velocity models.

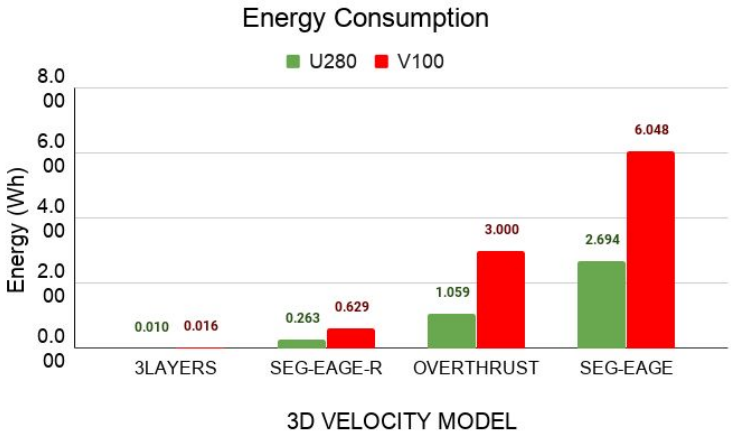


a) U280 FPGA; b) V100 GPU

Figure 5: 2D sections of the Shot Record generated for SEG-EAGE-R model at coordinates x=[107], y=[0-214], t=[0-2368].



a) Execution time (s)



b) Energy consumption (Wh)

Figure 6: Execution time (s) and energy consumption for 3D Seismic Modeling using U280 FPGA and V100 GPU.

5. Final Remarks

Comparing performance results between devices as different as FPGAs and GPUs can be tricky and perhaps inconclusive depending on what aspects are considered. NVidia claims that V100 GPU can reach up to 14 TFLOPS of peak performance with 250W max power consumption. On the other side, Xilinx's U280 can achieve 24.5 TOPS (INT8) peak performance with roughly the same power consumption (225W). This difference in nature of the peak performance information (32-bit TFLOPS on V100 and 8-bit TOPS for U280) makes it challenging to estimate the real gap between these two devices. However, if we were really optimistic and considered that one 32-bit floating-point operation could run as 4 INT8 operations, which is a huge approximation, V100 would have a nominal performance 2.3 times higher than U280.

That gap can be observed in the execution time measurements for the SEG-EAGE model in Figure 5. For this model, V100 executed 1.81 times faster than U280, but required 2.24 times more energy to produce the same Shot Record output. The energy gap was even higher with the medium-sized model OVERTHRUST, where V100 executed 1.36 times faster but requiring almost 3 times more energy.

Knowing that energy prices are a significant cost for HPC clusters operation, these results have shown that FPGA kernels can be a powerful alternative towards a *greener* production environment, possibly reducing operation costs while keeping high performance standards.

6. Demonstration Repository

The experiments presented on this white paper can be found on our demonstration repository on GitHub called rtm3d-demo (https://github.com/aluamorim/rtm3d-demo). The

repository includes the four velocity models used for benchmarking, the C++ Host program and the developed FPGA and GPU kernels. The repository also includes compilation and execution scripts for test automatization, along with a comprehensive README file containing a description of the repository structure and its usage.