Scientific Computing Accelerated on FPGA

FPGA acceleration of 3D CT reconstruction using OpenCL and oneAPI Tools

Daouda DIAKITE, PhD student

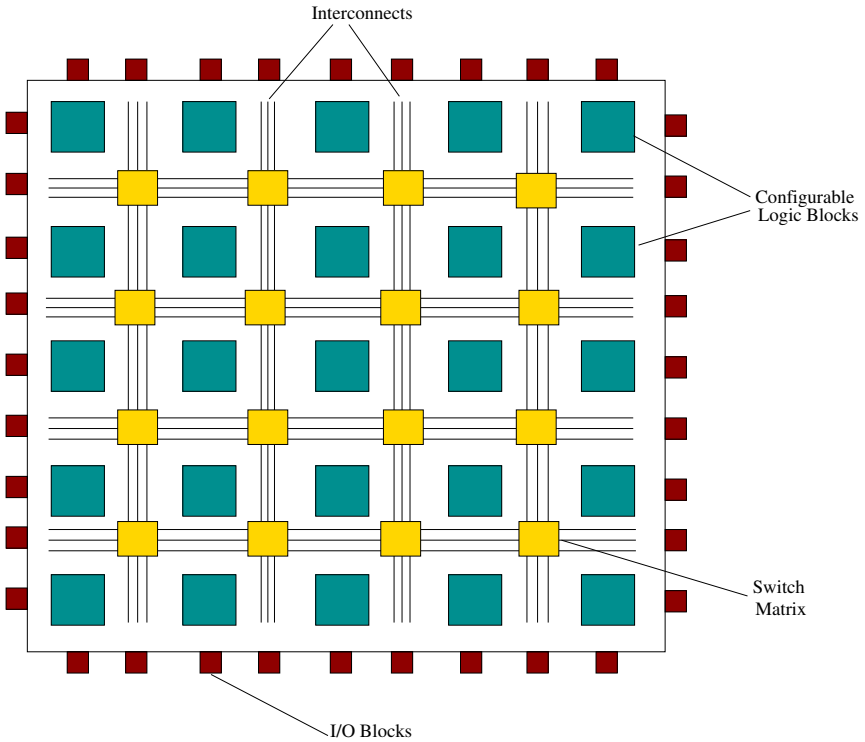
Université Paris-Saclay, CNRS, CentraleSupélec, L2S, 91190, Gif-sur-Yvette, France.

Supervisor: Nicolas GAC

July 04, 2022

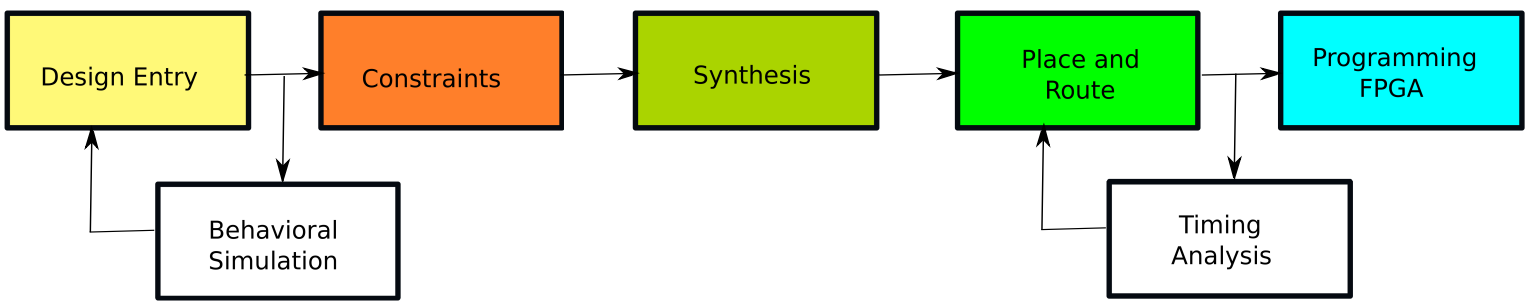
Field Programmable Gate Array (FPGA)

- Reconfigurable circuit
- With thousands of memory blocks and DSPs
- Millions of logic resources (FF, LUT, ...)



FPGA Design with HDL languages

- FPGAs build architectures
- Complex to program
- Through hardware description languages (HDL)
- FPGA design through HDL language is time-consuming



HLS tools

The two main FPGAs manufacturors have proposed their HLS tools :

- OneAPI and OpenCL SDK for Intel.
- Vitis and Vivado HLS for Xilinx.

Problematic

	2003	2005	2007	2009	2011	2013	2015	2017	2019
GPU			**10**	**17**	**18**	**9**	**13**	**20**	**30**
Nvidia			10	14	17	7	10	20	
AMD				1	1		1		
Intel(Larabee,Xeon phi)				2		2	2		
multi-GPU					3	3	3		1
CNN								2	13
Other processors	**2**	**3**	**10**	**7**	**3**	**2**	**1**	**2**	
CPU (MPI/OpenMP)	2	3	5	6	3	2	1	2	
Cell (IBM)			3						
DSP			2	1					
FPGA			**4**		**1**		**1**		

TABLE – Accelerators used in Fully3D conference.

FPGAs resurgence

- More built-in floating-point units (DSPs)
- Maturity of High-Level Synthesis (HLS)

Can FPGAs compete with GPUs for HPC applications ?

Tomographic reconstruction

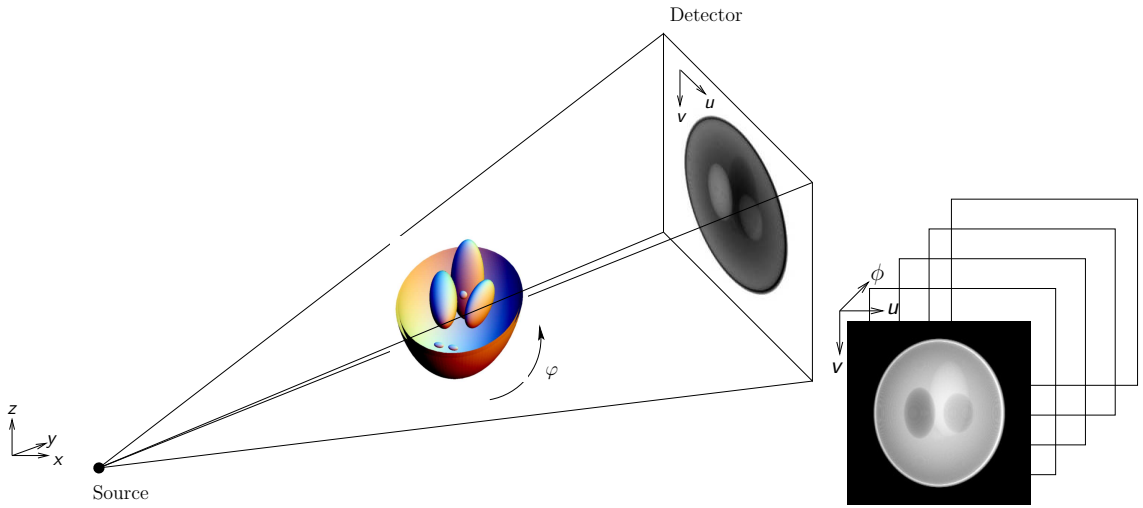


FIGURE – X-RAY CT Projection

Applications

- Non-Destructive testing
- Medical imaging

Iterative reconstruction algorithm

Inverse problem definition

$$g = Hf + \epsilon$$

The object f is obtained by minimizing the quadratic criterion :

$$\widehat{f} = arg\ min J_{MC} = arg\ min \frac{1}{2}||g - Hf||^2$$

Then iteratively : $f^{(n+1)} = f^{(n)} - \alpha.\nabla J_{MC}(f^{(n)})$

Problem size in tomography

- 1024^3 or 2048^3 volume cube to reconstruct.
- The size of system matrix is even larger (for 2048^3 H = 1 Exa Bytes).

Operators

- Matrix H and H^t are huge to be stored.
- Approach by operators : The forward and backward projections

Back-projector

- The operators are the forward and the backward projectors.
- In this presentation, we will only focus on the backward projection operator.

3D Back-projector

$$f(c) = \int g(u(\varphi, c), v(\varphi, c), \varphi).w(\varphi, c)^2 d\varphi$$

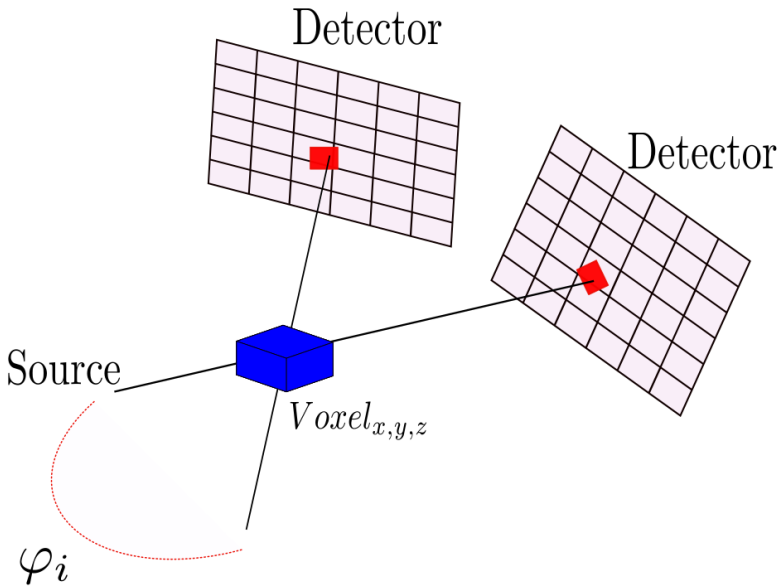
$$u(\varphi, c) = x * cos(\varphi) + y * sin(\varphi)$$

$$v(\varphi, c) = x * sin(sin\varphi) - y * cos(\varphi) + z$$

$c = (x, y, z)$ is voxel coordinates, $u(\varphi, c)$, $v(\varphi, c)$ detector cell

Back-projection algorithm

for all $Voxel_i$ **do**
  $voxel_{sum} \leftarrow 0$
  for all φ_j **do**
    $Compute(u, v)$
    $voxel_{sum} += projection[u, v, \varphi_j]$
  end for
  $volume[Voxel_i] = voxel_{sum}$
end for



How to improve performance ?

We need an algorithm architecture adequacy approach to better leverage the FPGA.

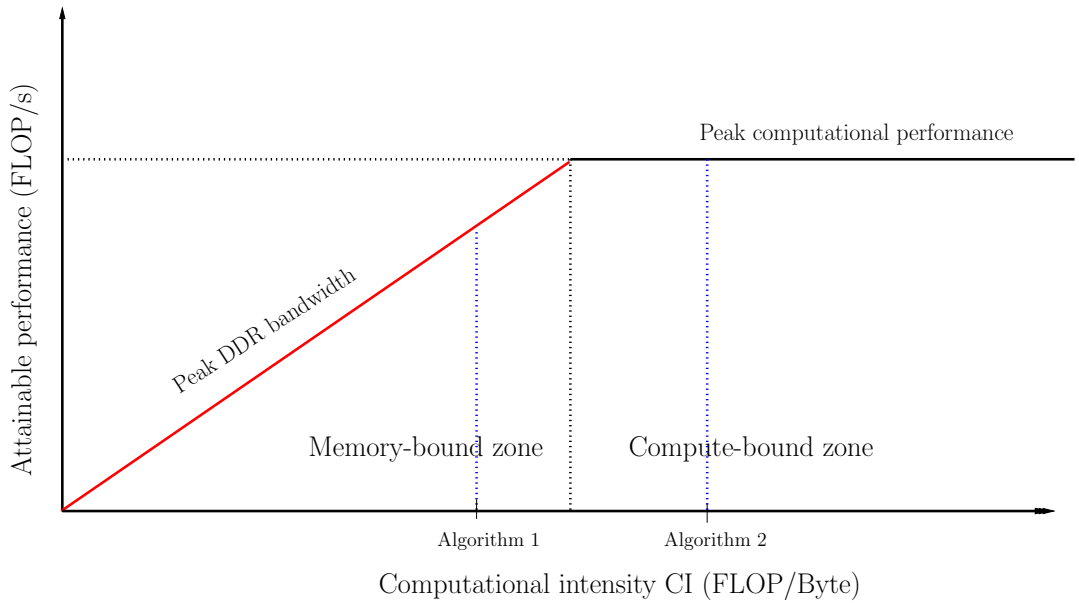
Algorithm side

- Perform an offline memory access analysis to make access pattern regular.
- Maximize the data reuse rate to reduce external memory access.

Architecture side

- Alleviate the memory bottleneck by maximizing local memory usage.
- Express high parallelism by using more DSP units.
- Use of Berkeley roofline model to characterize the algorithm.

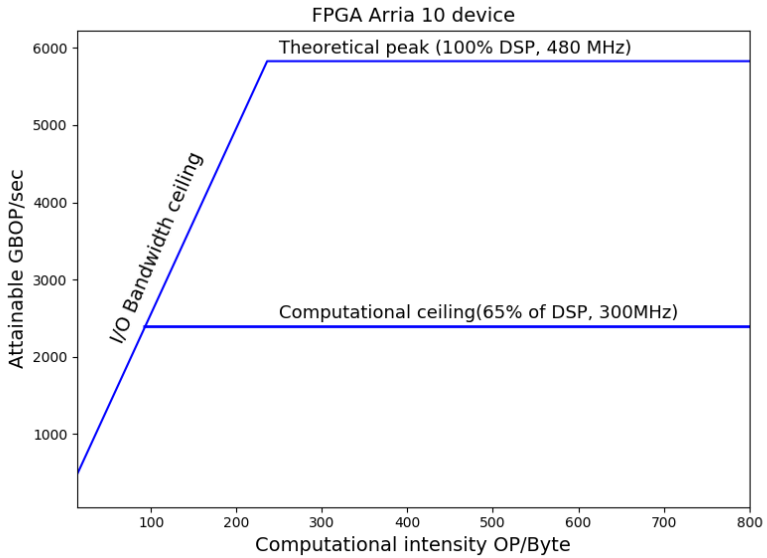
Berkeley Roofline [*Williams2009*]



- The roofline is a tool for visually and quickly observing the possible limitations of an algorithm.
- The model is based on two keys parameters :
  - The device computational peak performance
  - The attainable bandwidth

$$Attainable\ Performance(FLOP/s) = min(CP, AI \times BW).$$

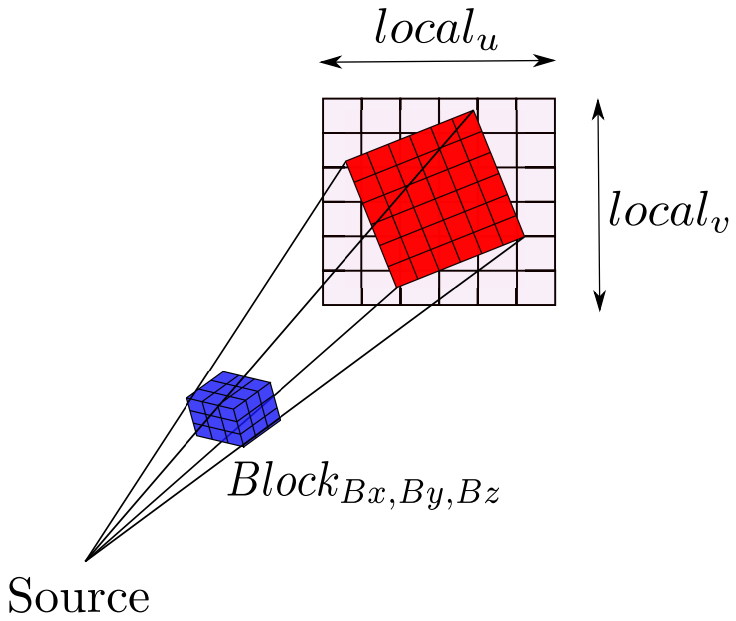
Roofline for FPGAs [*Silva2013*]



Considerations in the extended model

- Operations : On FPGAs, we cannot only consider floating-point operations because there are other alternative such as fixed point operations.
- Computational ceiling : The architecture of an FPGA design is not fixed, so the computational roof is influenced by the resource consumption.
- Bandwidth : multiple bandwidths (PCIe, Network, ...) to take into account.
- Scalability : This is defined by the level of parallelism introduced by loop unrolling or replication of the kernel.

Memory access strategy



$$local_u = \sqrt{B_x^2 + B_y^2}$$

$$local_v = \sqrt{2} * B_z$$

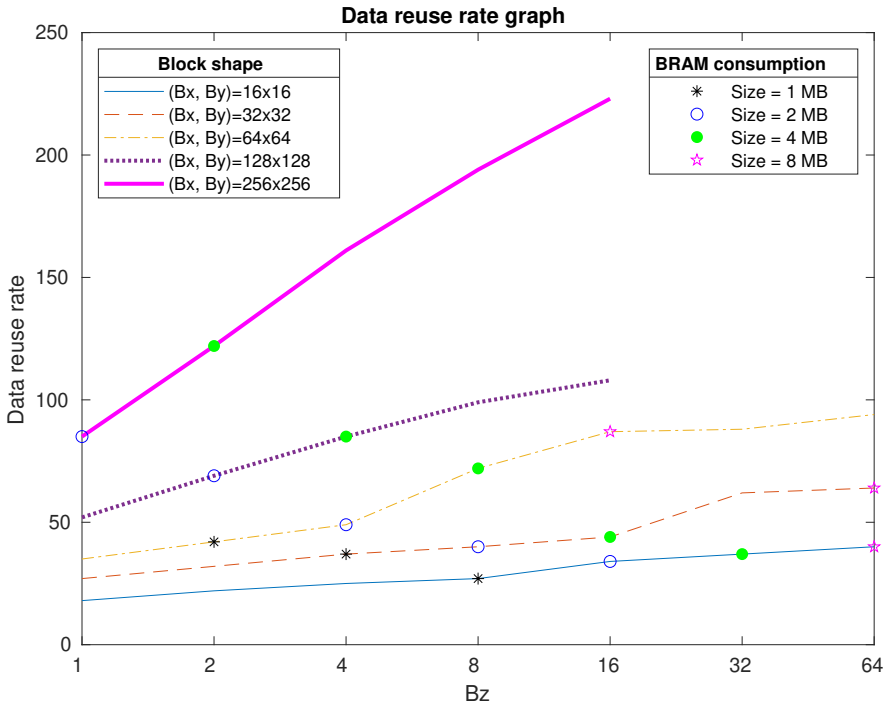
$$Reuse = \frac{B_x * B_y * B_z}{\#Memory\ I/O}$$

Block size choice

One should choose a size of block that :

- Maximize the data reuse rate.
- Optimize the BRAM usage.
- Reduce access to global memory.

Data reuse rate



Method advantage

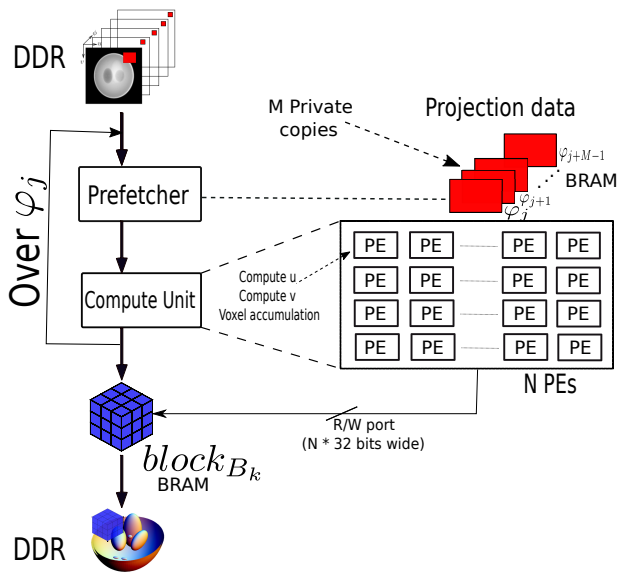
- We have two important parameters : Data reuse rate and BRAM consumption.
- For the same BRAM consumption, the less the block is thick the more the reuse.

Roofline model for back-projector

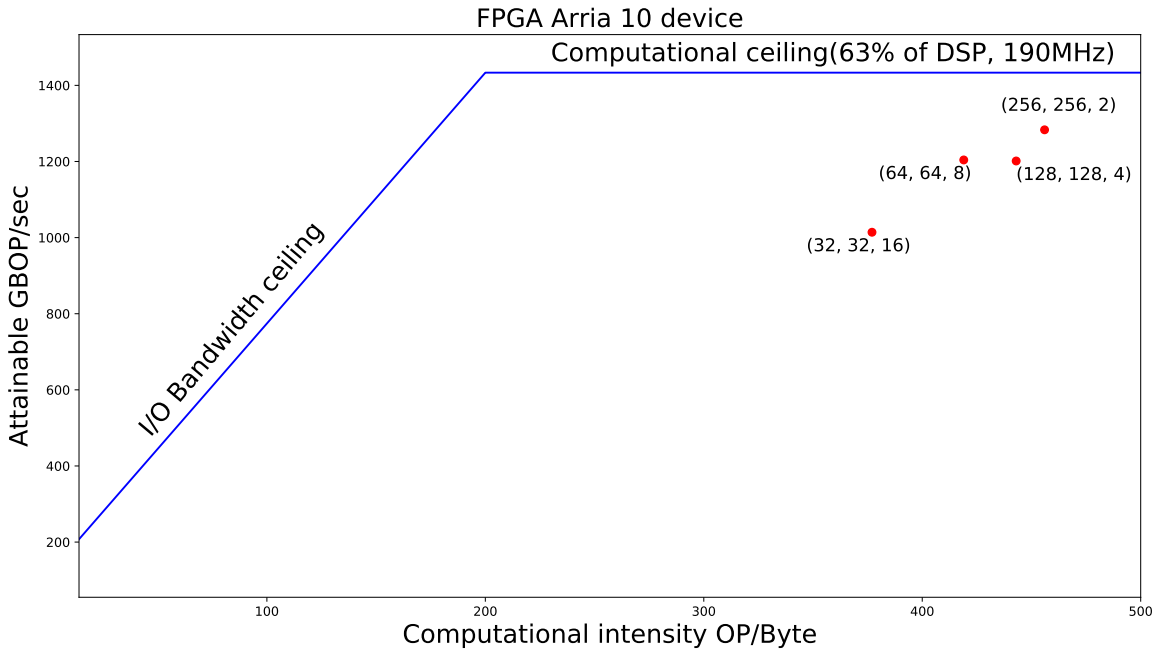
	B_z	#Operations (GBOP)	#Memory accesses (GB)	CI
BP-cache	N/A	236	17.2	13.7
BP-prefetch	$64 \times 64 \times 1$	253	1.1	236
	$32 \times 32 \times 16$	253	0.67	377
	$64 \times 64 \times 8$	253	0.6	419
	$128 \times 128 \times 4$	253	0.57	443
	$256 \times 256 \times 2$	253	0.55	456

Architecture on FPGA

- The BP-Prefetch architecture on Arria 10 device.
- Pipeline with N=64 PEs.



Roofline model for back-projector



Reference	$32^2 \times 16$	$64^2 \times 8$	$128^2 \times 4$	$256^2 \times 2$
Stall	0.2	0.06	0.56	0.06
Occupancy	74.7	84.1	90.2	94

Results on Arria 10

Reference	Platform	Volume	Acc. /voxel	Time (s)	Freq (Mhz)
BP-cache	Arria 10	256^3	256	3.62	150
BP-prefetch $256^2 \times 2$	Arria 10	256^3	256	0.396	180

TABLE – Performance comparison of our work and other works

Resources consumption

	BP-cache	BP-prefetch
DSP	406 (27%)	949 (62%)
BRAM	1758 (65%)	1952 (73%)

TABLE – Arria 10

Scalability to Stratix 10

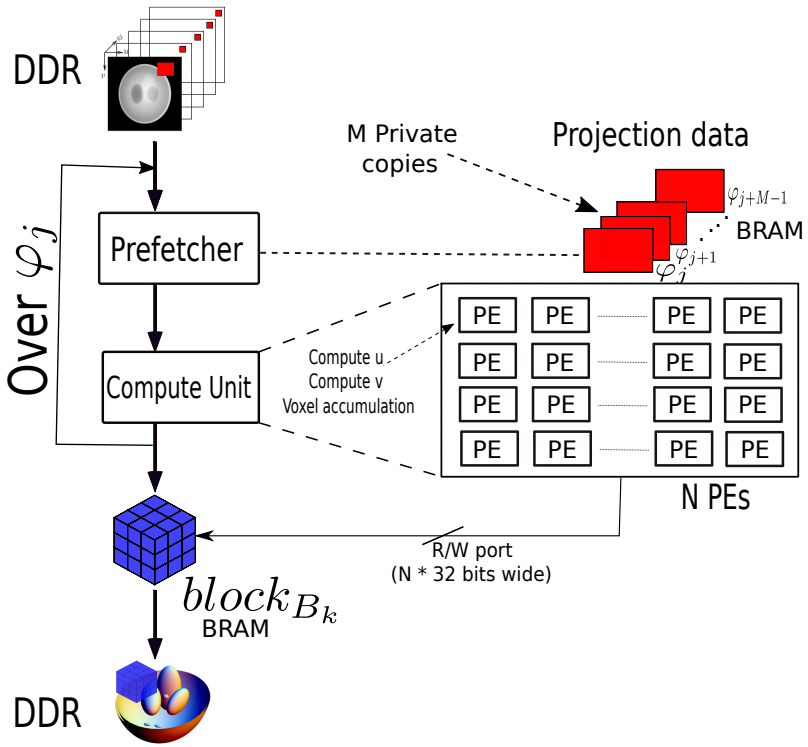
- To achieve even more throughput, we port our design on Intel Stratix 10 device.
- The Stratix 10 provides more compute capability than the Arria 10 device.
- For our pipeline architecture, we are able to scale up 256 PEs on the Stratix 10.

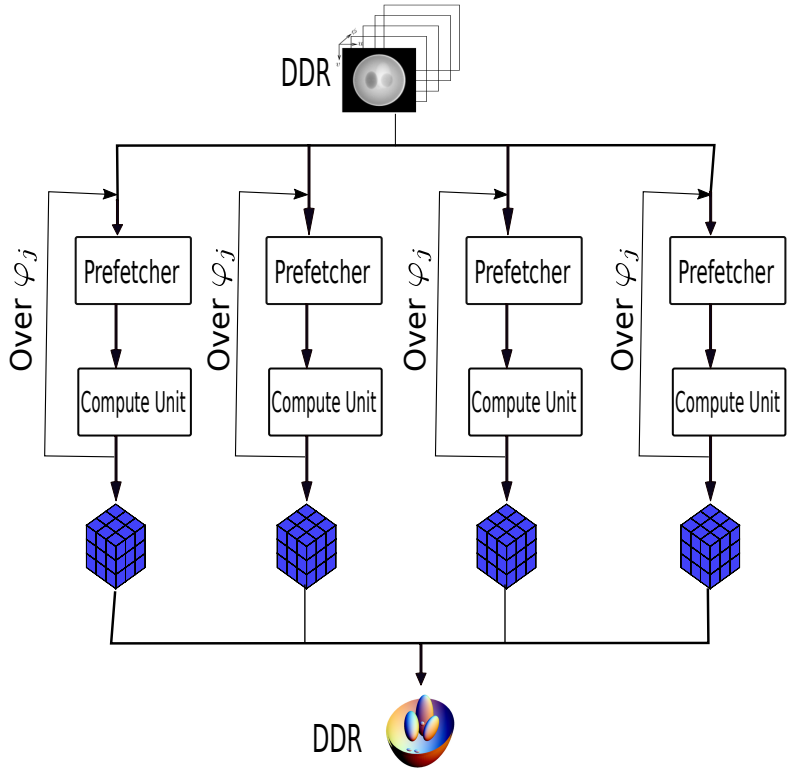
Parallelism

Two strategy to implement on Stratix 10 :

- Single kernel
- Multikernel

Scalability to Stratix 10





Single vs. multikernel

- Single kernel : $N = 256$ PEs with one kernel
- Multikernel : : $N = 64$ PEs with 4 kernels

Stratix 10 results

Design	Block size	BRAM	DSP	Freq (MHz)	Time (s)
Single kernel	$256^2 \times 2$	5881 (50%)	2845 (49%)	117	0.24
Multikernel	$64^2 \times 8$	3898 (33%)	3282 (57%)	172.5	0.12

TABLE – Single kernel versus multikernel on Stratix 10

Design efficiency

- The occupancy of the single kernel version was not very high due to memory latency and the kernel low frequency.
- The multikernel version has considerably reduced the BRAM consumption.
- Kernel replication introduces extra logic utilization.

Comparison with CPU and GPU

CPU, GPU, FPGA

Device	Power(W)	Energy(mWh)	Execution time(s)
CPU E5-2667	47	862	66
Titan X Pascal	237	0,92	0.014
Jetson TX2	12,9	0,91	0,253
Arria 10	14,9	1.7	0.42
Stratix 10	27	0.9	0.12

Tomography acceleration

- FPGA Stratix 10 is more efficient than embedded GPU.
- The HPC GPU is 10× faster than Stratix 10.
- The energies consumed are equivalent for the Stratix and the HPC GPU.
- In term of worst-case power consumption, FPGA Stratix 10 is more efficient than HPC GPU.

Pipeline efficiency

- We evaluated the pipeline efficiency for each device.
- It represents the number of cycle needed to update one voxel for the pipeline or the CUDA core.

$$\eta = \frac{Time * f_{max} * \#PE}{\#Accumulations}$$

where

$Time$ is the kernel execution time (s)

f_{max} is the operating frequency

$\#PE$ is the number of CUDA cores or PEs

$\#Accumulations$ is the total number of voxel updates.

Pipeline efficiency

Pipeline efficiency

	GPU Titan X	FPGA Arria 10 BP-Cache	FPGA Arria 10 BP-Prefetch
Operations (op)	256^4	256^4	256^4
Cores	3584 (CUDA cores)	32 (PEs)	64 (PEs)
Frequency (MHz)	1481	150	189
Runtime (s)	**0.014**	5.34	0.42
cycle/op/core	17.3	5.97	**1.18**

Voxel update

- FPGA OpenCL : 1.18 clock cycles per PE for Arria 10
- FPGA OpenCL : 1.23 cycle per PE for Stratix 10
- GPU : 17 cycles per CUDA core
- FPGA VHDL : 1 cycle per PE [*Gac2008*]

Results comparison

Reference	Platform	Volume	Acc. /voxel	Time (s)	GUPS	GUpdate /cycle /MAC
Choi2016	4×Virtex-6	$512^2 \times 372$	831	3.7	20.4	0.156
Wen2020	ZCU102	$1024^2 \times 128$	502	2.10	29.9	0.073
GPU	Jetson TX2	256^3	256	0.25	15.8	0.078
GPU	Titan X Pascal	256^3	256	0.014	285	0.071
Our work	Stratix 10	256^3	256	0.12	33.3	0.063

TABLE – Performance comparison of our work and other works

Comparison

- Efficient DSP usage compared to other works
- Use of OpenCL instead of Vivado HLS

[Choi2016] Y.-k. Choi et al., "Acceleration of EM-based 3D CT reconstruction using FPGA," in IEEE TBioCAS, vol. 10, no. 3, 2016, pp. 754–767.
[Wen2020] S. Wen et al., "FPGA-accelerated automatic alignment for three-dimensional tomography," in 2020

Comparison with ONEAPI

- Programming across different architecture using one single language DPC++
- Easy to use for software and hardware programmers
- Use of FPGA specific pragmas and attributes to optimize the design
- Auto-detect accelerator architecture during application runtime
- Single file for kernel and host applications.

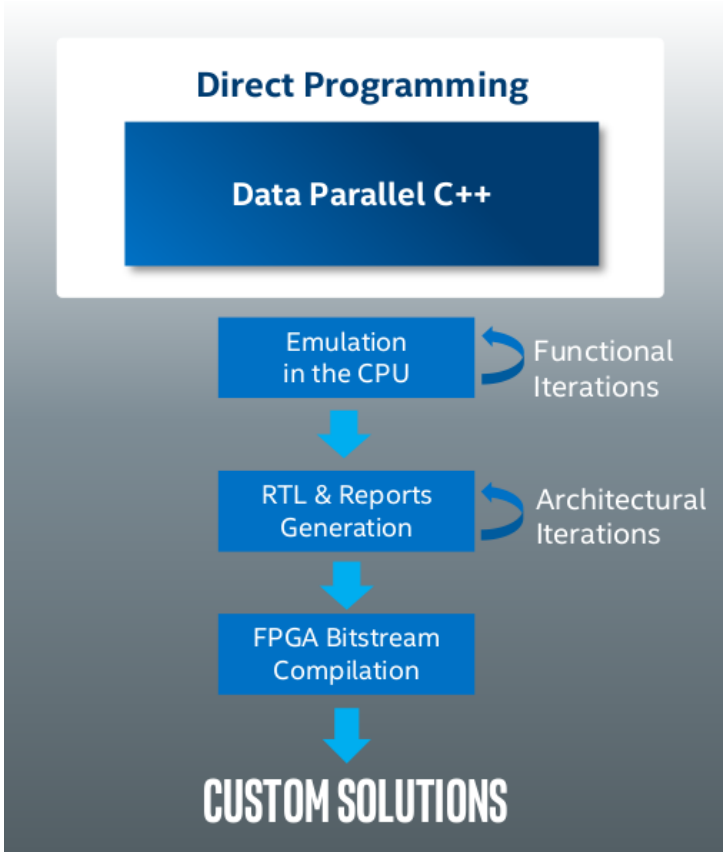


Image source : Intel

Comparison with ONEAPI

ONEAPI first results

- We have implemented the BP-Cache and the BP-Prefetch designs using Intel ONEAPI on Arria 10

Design	Tool	BRAM	DSP	Freq (MHz)	Time (s)
BP-Cache	OpenCL	1758 (65%)	406 (27%)	150	5.34
	OneAPI	2088 (77%)	374 (25%)	163	6.43
BP-Prefetch	OpenCL	1952 (71%)	949 (63%)	189	0.42
	OneAPI	1041 (38%)	878 (58%)	229	0.54

- About 20% of performance loss compared to OpenCL.
- In term of development effort and number of lines of code oneAPI is way advantageous.

Conclusion for 3D back-projector

Conclusion

- The 3D back-projection has been accelerate on FPGA using OpenCL. This acceleration took advantage of FPGA on-chip BRAM to achieve better performance.
- To stand up in front of GPU, FPGA need some hardware improvement and designer expertise for tomography acceleration.
- However, HLS tools are still improving making more suitable for specific optimization.
- We also have hardware improvements for FPGAs as well :
  - The new memory HBM2 to respond to memory bottleneck for example.
  - The Intel Agilex family offering more computational power and more energy efficiency than Stratix 10.

Perspectives

Tools comparison

- Perform tools comparison between Intel HLS tools.
- Deepen the work on the Intel ONEAPI.

FPGA in tomography

- Apply this methodology to the projection operator

FPGA in radioastronomy

- FPGAs are widely used in radioastronomy instrument and correlator
- FPGAs are being explored for imaging by ASTRON team
  - *Veenboer2019, Corda2022*
- Data-flow processing and energy efficiency
- Consideration of the deconvolution algorithm.

Thanks !

Questions ?