list
CEA tech

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Soutenance de thèse de doctorat

UNIFICATION DES MÉMOIRES RÉPARTIES DANS LES SYSTÈMES HÉTÉROGÈNES

Soutenue par: **Erwan Lenormand** le 31 janvier 2022

Composition du jury

Président:	**M. Marc Pérache**, CEA DAM, Université Paris-Saclay
Rapporteurs:	**M. Tanguy Risset**, CITI, INSA-Lyon **M. Samuel Thibault**, INRIA, Université de Bordeaux
Examinateurs:	**Mme Catherine Dezan**, Lab-STICC, Université de Bretagne Occidentale **M. Olivier Muller**, TIMA, Université Grenoble Alpe **Mme Soraya Zertal**, LI-PARAD, Université Paris-Saclay
Directeur de thèse:	**M. Henri-Pierre Charles**, CEA LIST, Université Grenoble Alpes

Membres invités

Co-encadrants:	**M. Loïc Cudennec**, DGA MI **M. Stéphane Louise**, CEA LIST, Université Paris-Saclay

Membre externe

Co-encadrant:	**M. Thierry Goubier**, Ministère de l'Europe et des Affaires étrangères

TOWARDS MORE HETEROGENEITY IN COMPUTER SYSTEMS
From supercomputer to System on Chip

Supercomputer

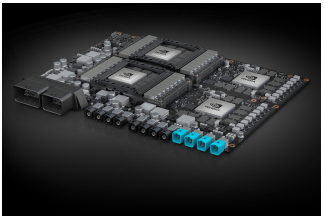
© CCS, University of Tsukuba

Cygnus supercomputer

CPU-GPU-FPGA

2019

Embedded system



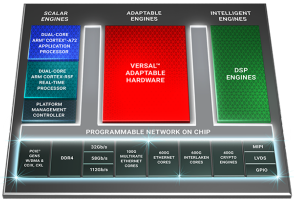
© Nvidia

Nvidia Drive AGX Pegasus

CPU-GPU

2018

System on Chip



© Xilinx

Xilinx Versal

CPU-FPGA

2020

OUTLOOK: 1 - CONTEXT & MOTIVATION

1. Context & motivation
2. State-of-the-art
3. Contributions & Experimentation
4. Conclusion & Outlook

EVOLUTION OF INTEGRATED CIRCUITS

End of Dennard scaling has led to heterogeneous systems

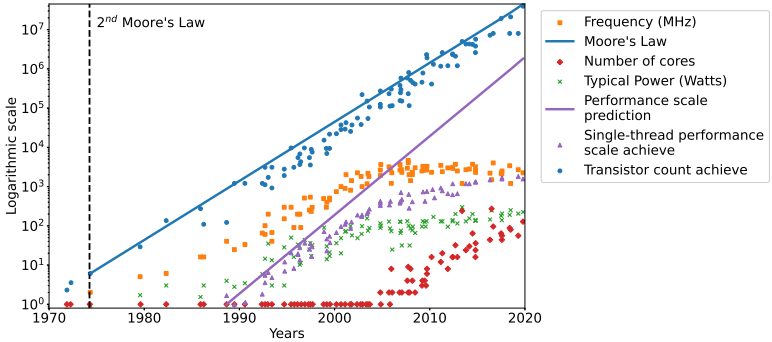


Figure 1: 48 Years of Microprocessor Trend Data.
Original data up to the year 2010 collected by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten New data collected for 2010-2019 by K. Rupp

Moore's revised law (1975)
The count of integrated transistors on a chip doubles every two years

+

~~**Dennard's MOSFET scaling**~~
~~Thanks to miniaturization, IC power density stays constant~~

→

~~**David House's interpretation**~~
~~Computer chip performance would double every 18 months~~

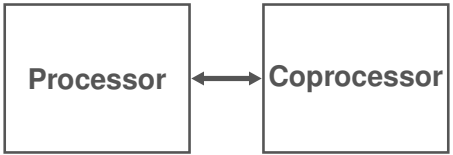
VARIOUS SOURCES OF HETEROGENEITY

What kind of system is targeted in this work?

Heterogeneity in a computing system

heterogeneity comes from integrating components with the same function but without the same architecture

- Storage
- Random-access memory
- Instruction set architecture
- **Processor(s) coupled with coprocessor(s)**



ARCHITECTURALLY TRULY DIVERSE PROCESSING UNITS

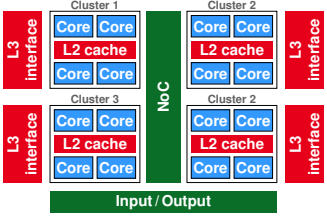


Figure 2: Central Processing Unit (CPU)

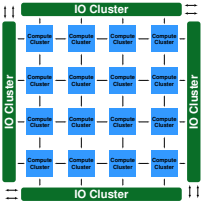


Figure 3: Manycore processor

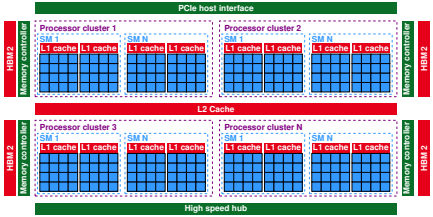


Figure 4: Graphics processing unit (GPU)

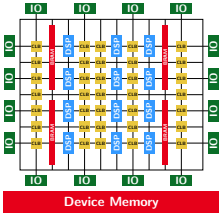


Figure 5: Field-programmable gate array (FPGA)

A complex and time-consuming task

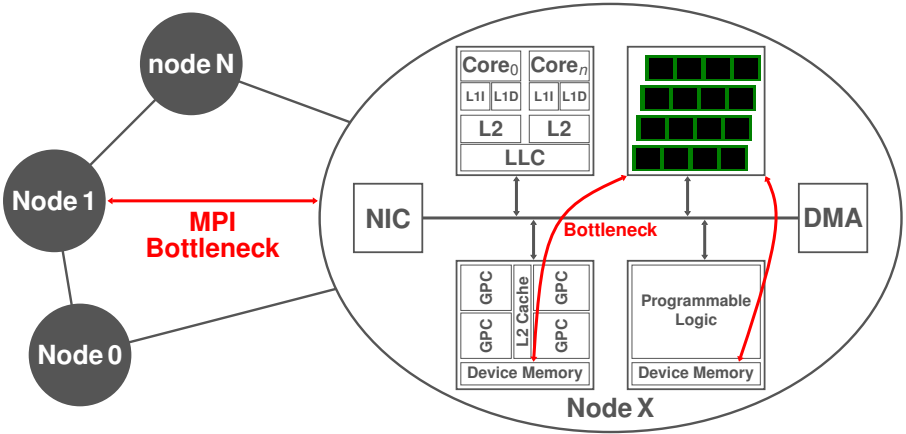


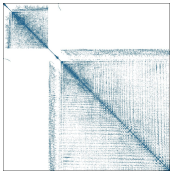
Figure 6: Distributed heterogeneous system: More hybrid programming model

SUITABILITY OF PROCESSING UNITS FOR SCIENTIFIC WORKLOADS

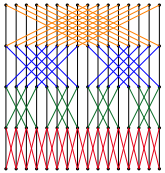
Six dwarfs of High-Performance Computing

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,j} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,j} & \cdots & a_{2,n} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,j} & \cdots & a_{n,n} \end{pmatrix} \begin{pmatrix} x_{1,1} \\ x_{2,1} \\ \vdots \\ x_{n,1} \end{pmatrix} = \begin{pmatrix} c_{1,1} \\ c_{2,1} \\ \vdots \\ c_{n,1} \end{pmatrix}$$

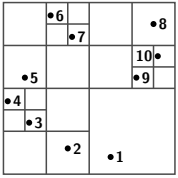
1 - Dense Linear Algebra



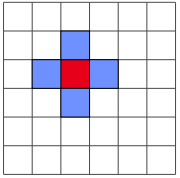
2 - Sparse Linear Algebra



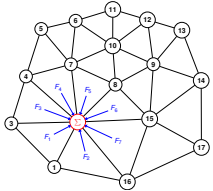
3 - Spectral Methods



4 - N-Body Methods



5 - Structured meshes



6 - Unstructured meshes

Originally 7 dwarfs were identified by Phillip Colella (Berkeley) in 2004

SUITABILITY OF PROCESSING UNITS FOR SCIENTIFIC WORKLOADS

Six dwarfs of High-Performance Computing

Dwarf	Processing speed	Energy efficiency
Dense Linear Algebra	**GPU**	**GPU**
Sparse Linear Algebra	**GPU**	**FPGA**
Spectral Methods	**GPU≈FPGA**	**FPGA**
N-Body Methods	**GPU**	**FPGA**
Structured meshes	**GPU≈FPGA**	**FPGA**
Unstructured meshes	**FPGA**	**FPGA**

Tableau 1: Suitability Analysis of coprocessors for scientific computing algorithms

OUTLOOK: 2 - STATE-OF-THE-ART

1. Context & motivation
2. State-of-the-art
3. Contributions & Experimentation
4. Conclusion & Outlook

DISTRIBUTED MEMORY MODELS
Overview

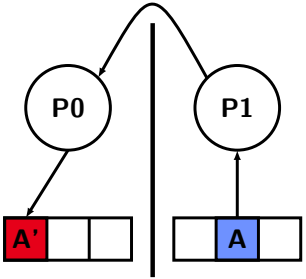


Figure 7: Message-passing

- MPI
- Bulk DMA transfers

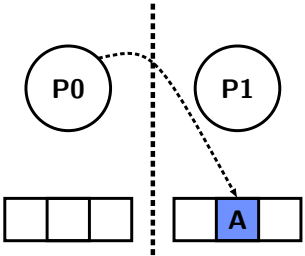


Figure 8: Partitioned-memory

- PGAS Languages
- PGAS API

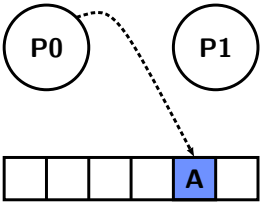
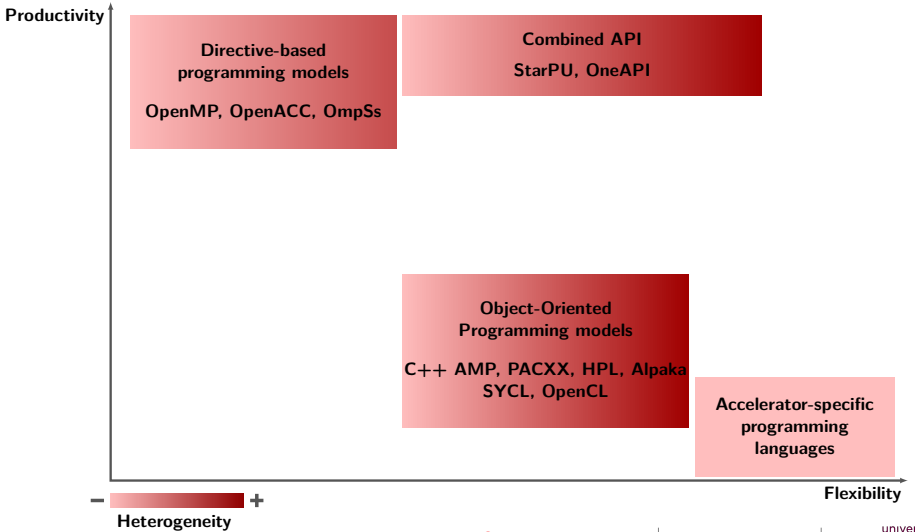


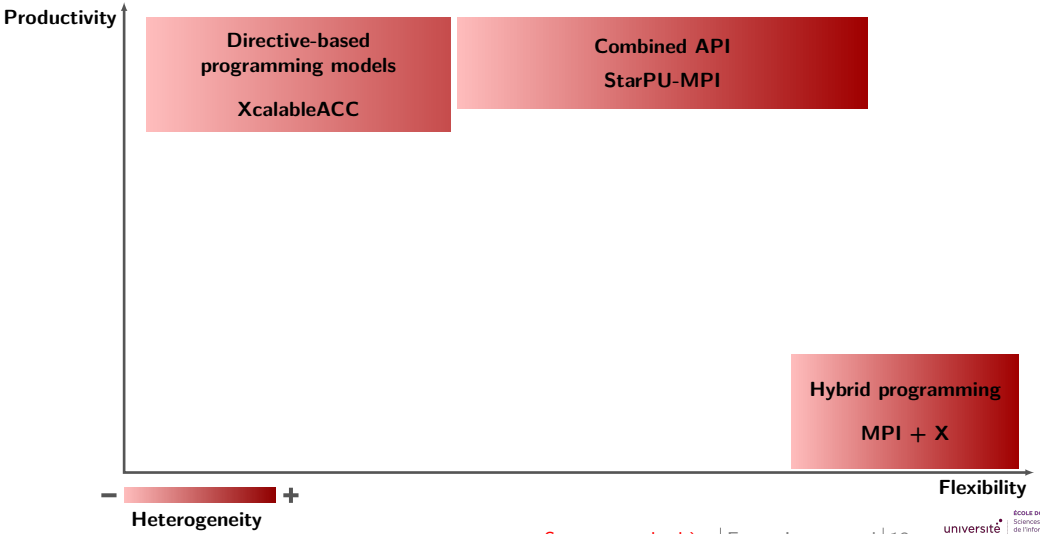
Figure 9: Shared-memory

- Cache-Coherent Network on Chip
- Software-Distributed Shared Memory
- Coherent interconnect for coprocessors

STATE OF THE ART
Programming heterogeneous node

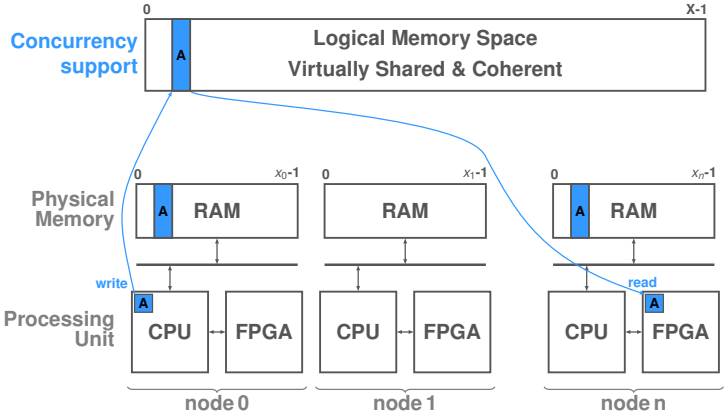


STATE OF THE ART
Programming distributed heterogeneous nodes



WORK POSITIONING

Unifying distributed memories in a heterogeneous system with FPGA



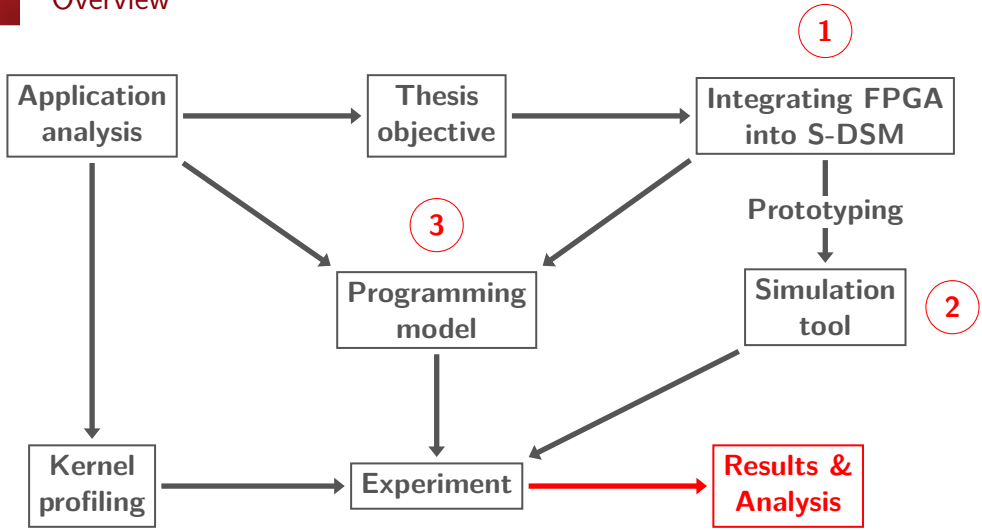
Erwan Lenormand et al. "Unification des mémoires réparties dans un système hétérogène avec accélérateur reconfigurable : exposé de principe". In: *Conférence d'informatique en Parallélisme, Architecture et Système*. 2019

OUTLOOK: 3 - CONTRIBUTIONS & EXPERIMENTATION

1. Context & motivation
2. State-of-the-art
3. Contributions & Experimentation
4. Conclusion & Outlook

THESIS CONTRIBUTIONS

Overview



SOFTWARE-DISTRIBUTED SHARED MEMORY

Overview of the API used in this work

- API named SAT (Share Among Things), developed at CEA since 2017
- Clients run user code & Servers run API code
- Shared data is sliced into atomic chunks

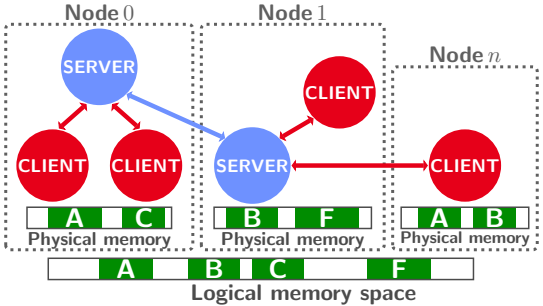


Figure 10: Topology of SAT

SOFTWARE-DISTRIBUTED SHARED MEMORY

Allocation and access to shared data

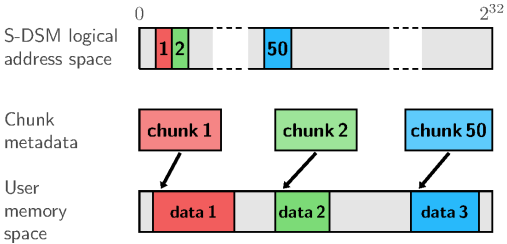


Figure 11: Chunks allocation

```
1   SAT_Chunk_t * chunk = NULL;
2   uint32_t ID = 1, size = 197;
3
4   /* Chunk allocation */
5   chunk = SAT_MALLOC(ID, size);
6
7   /* Access to chunk */
8   _SAT_WRITE(chunk);
9   foo(chunk->data);
10  _SAT_RELEASE(chunk);
```

INTEGRATING FPGA INTO S-DSM

Breaking master-slave model

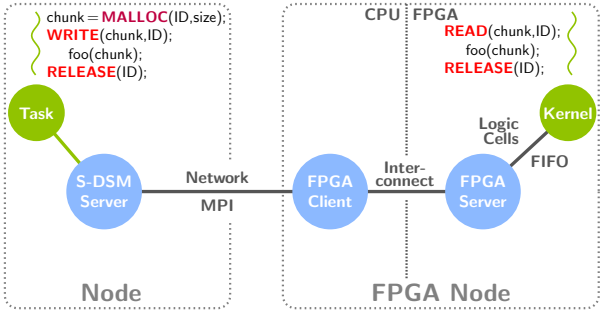


Figure 12: Integration topology

Goal

- Accelerators initiate remote memory access
- Supporting concurrent distributed data access
- Targeting irregular compute kernels

INTEGRATING FPGA INTO S-DSM
complex prototyping work

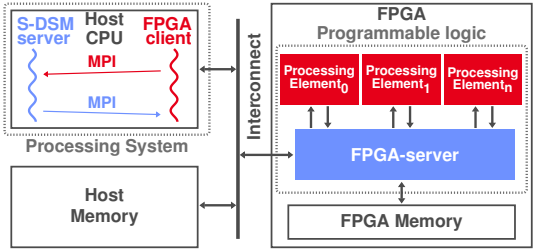


Figure 13: Overview of the reconfigurable accelerator integration architecture

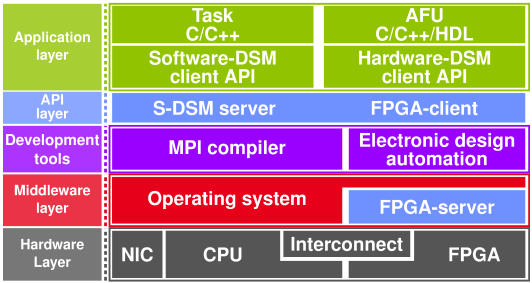
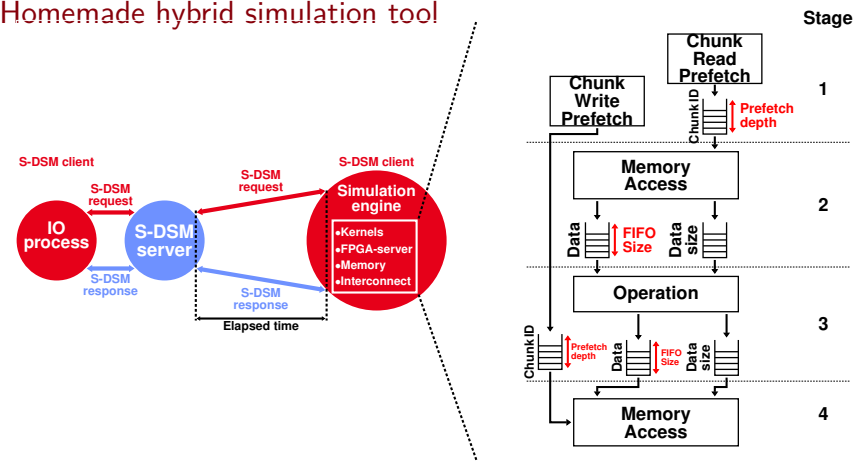


Figure 14: Overview of the Hardware/software stack

PRE-PROTOTYPING PERFORMANCE EVALUATION

Homemade hybrid simulation tool



Erwan Lenormand et al. "A combined fast/cycle accurate simulation tool for reconfigurable accelerator evaluation: application to distributed data management". In: *International Workshop on Rapid System Prototyping*. 2020

SIMULATION PLATFORM

- In-house micro-cluster of heterogeneous development boards and a high-performance gateway

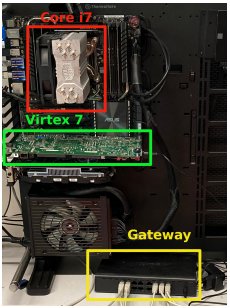


Figure 15: Main node

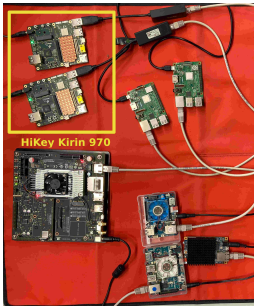


Figure 16: Heterogeneous nodes

Node	Processor	Latency		
		FPGA ping-pong	S-DSM read	S-DSM write
Local	Intel Core i7 6800K	20–24 µs	≈ 360 µs	≈ 200 µs
Remote	Arm Cortex A73/A53		≈ 1300 µs	≈ 600 µs

Tableau 2: Simulation platform description

OUTLOOK: 3 - CONTRIBUTIONS & EXPERIMENTATION

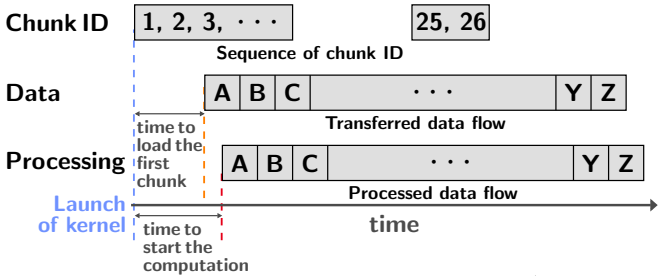
PROGRAMMING MODEL
Main idea

Goal

- Abstract the irregular shape of data structure
- Limit the number of data access requests
- Hide memory access latency

Principle

- Use chunks metadata
- Slice chunks to access granularity
- Improve data locality
- Access chunks as stream

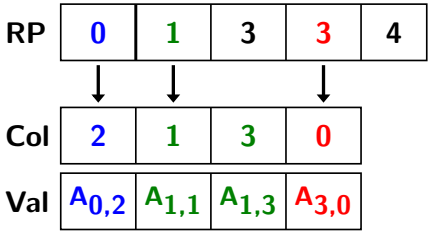


OUTLOOK: 3 - CONTRIBUTIONS & EXPERIMENTATION

CHUNKS STRUCTURE
Compressed Sparse Row Format adaptation

$$\begin{bmatrix} 0 & 0 & A_{0,2} & 0 \\ 0 & A_{1,1} & 0 & A_{1,3} \\ 0 & 0 & 0 & 0 \\ A_{3,0} & 0 & 0 & 0 \end{bmatrix}$$

Dense format



Compressed sparse row format

Chunk adaptation

- Abstracting irregularity through metadata
- Improving locality by grouping data

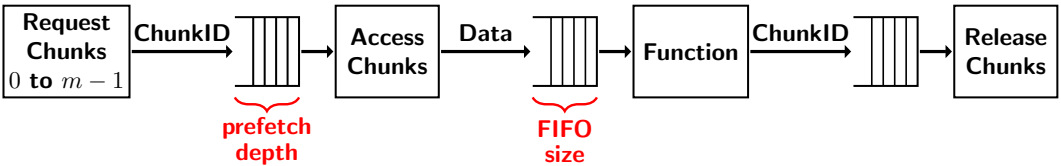
ID	count	
0	1	$(2,A_{0,2})$
1	2	$(1,A_{1,1})$ $(3,A_{1,3})$
3	1	$(0,A_{3,0})$

Chunk Compressed sparse row format

CHUNK COMPRESSED SPARSE ROW FORMAT TRAVERSAL

```
for(row=0; row < m; ++row)
{
  chunk = _SAT_LOOKUP(row);
  _SAT_READ(chunk);
  for(i=0;i < chunk->size/2; i+=2)
    f(chunk->data[i],chunk->data[i+1]);
  _SAT_RELEASE(chunk);
}
```

Code 1: Chunk CSR traversal



CASE STUDY I: SPGEMM
Compute kernel

Algorithm Gustavson's row-wise matrix multiplication algorithm.

$A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{n \times p}$, $C \in \mathbb{R}^{m \times p}$

for i = 0,...,m-1 **do**
  For Each $A_{i,k}$ in row $A_{i,*}$ **do**
    For Each $B_{k,j}$ in row $B_{k,*}$ **do**
      if $C_{i,j} == 0$ **then**
        $C_{i,j} = A_{i,k} \times B_{k,j}$
      else
        $C_{i,j}$ += $A_{i,k} \times B_{k,j}$

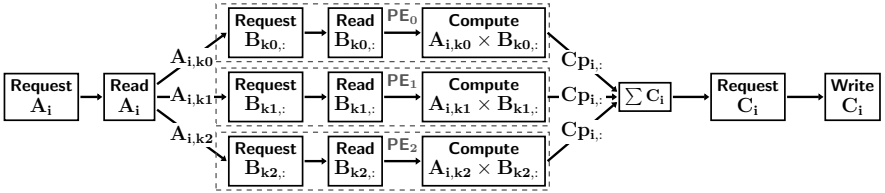
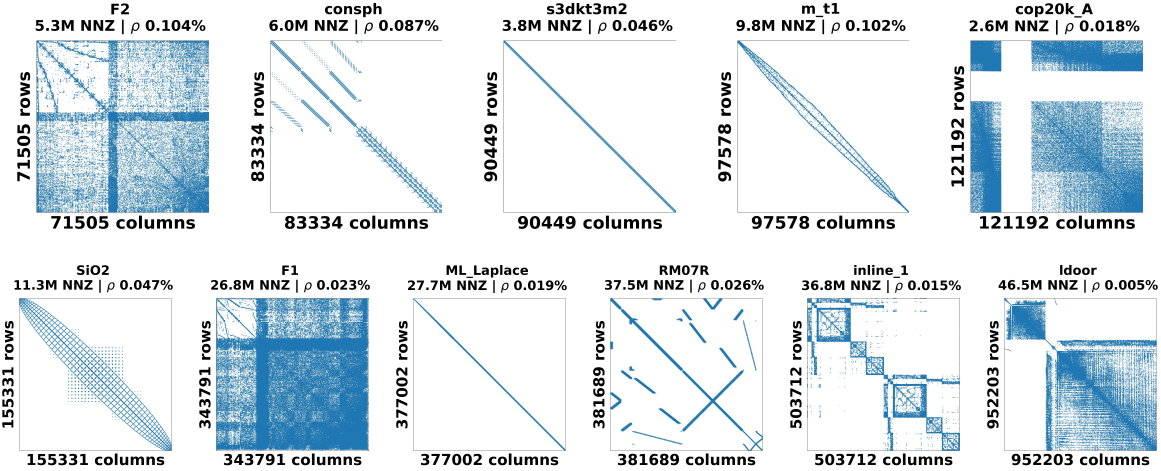


Figure 17: Dataflow overview of the compute kernel with 3 PEs.

CASE STUDY I: SPGEMM
Matrix dataset



Square matrix dataset from the University of Florida Sparse Matrix Collection

CASE STUDY I: SPGEMM

Performance according to the number of processing elements

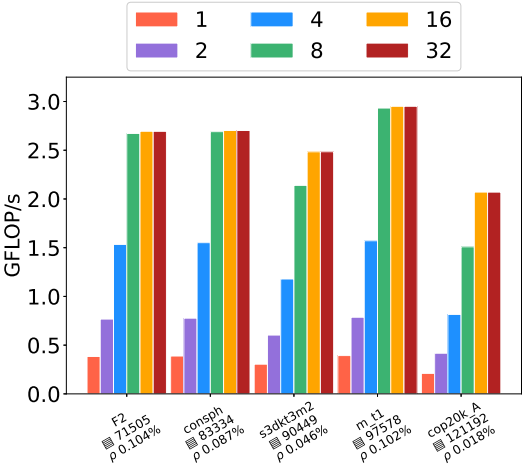


Figure 18: Computation speed in GFLOP/s according to the number of processing elements.

Scaling failure

- No speed-up beyond 8 PEs
- Why? Data starvation!

CASE STUDY I: SPGEMM

Performance according to the number of processing elements

Explanation

- Local memory bandwidth bottleneck
- Remote data access latency is successfully masked

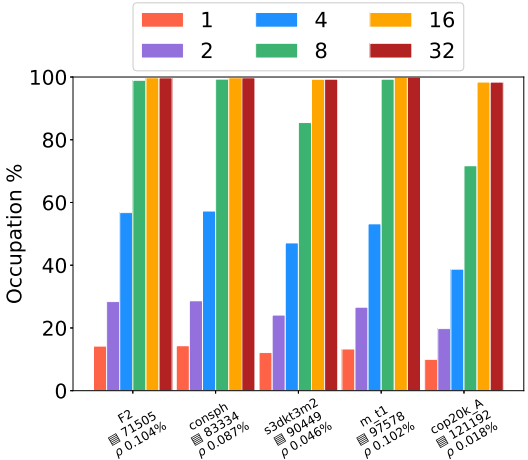


Figure 19: Memory controller activity (occupancy percentage) according to the number of processing elements.

CASE STUDY I: SPGEMM
Performance according the system topology

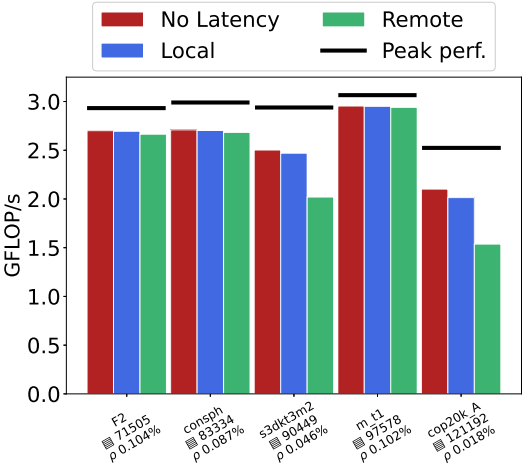


Figure 20: Computation speed in GFLOP/s according the system topology.

Analysis

- Impact of higher data access latency:
  - The densest matrices are unaffected
  - The sparsest matrices are affected
- Theoretical peak performance is not reached (but almost)

CASE STUDY I: SPGEMM
Performance according the system topology

Explanation

- Data starvation:
  - For the densest matrices: Local memory bandwidth bottleneck
  - For the sparsest matrices: Remote data access latency
- The width of the intern data bus is not fully exploited

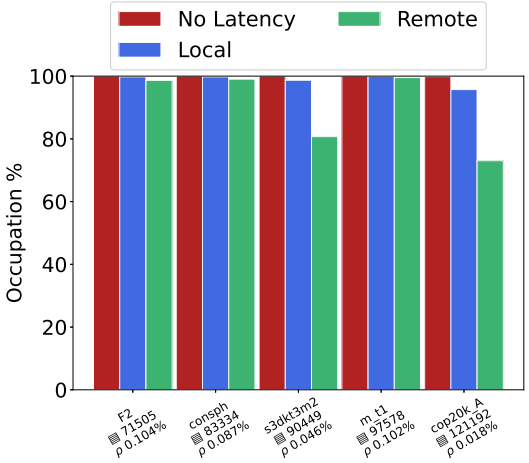


Figure 21: Memory controller activity (occupancy percentage) according the system topology.

CASE STUDY I: SPGEMM

Corroborate the analysis with the set of large matrices

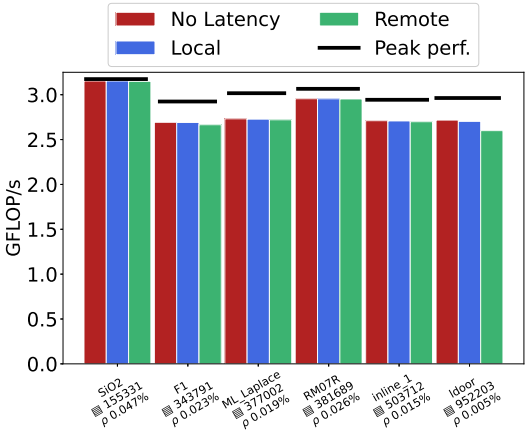


Figure 22: Computation speed in GFLOP/s according the system topology.

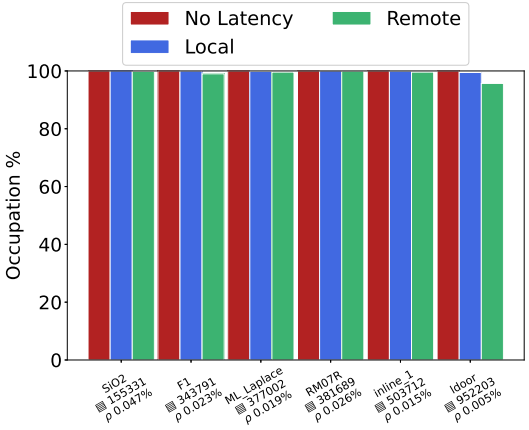


Figure 23: Memory controller activity (occupancy percentage) according the system topology.

CASE STUDY I: SPGEMM
Discussion

- A data starvation is caused by a local memory bandwidth bottleneck
- Data access latencies can be masked when the workload is large enough
- The data bus is not fully exploited due to the irregularity of the rows
- The programming model is well suited for large datasets

OUTLOOK: 3 - CONTRIBUTIONS & EXPERIMENTATION

1. Context & motivation
2. State-of-the-art
3. Contributions & Experimentation
   - Integrating FPGA into S-DSM
   - Simulation tool
   - Programming model
     - Sparse Linear Algebra
     - Unstructured mesh processing
4. Conclusion & Outlook

UNSTRUCTURED MESH
Illustration

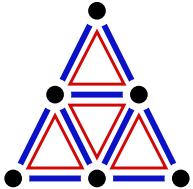


Figure 24: Representation of a mesh composed of vertices, nodes and edges

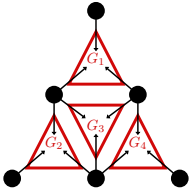


Figure 25: Example of a mesh update operation

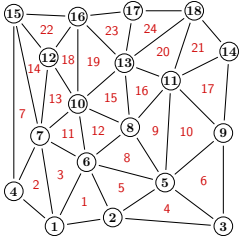
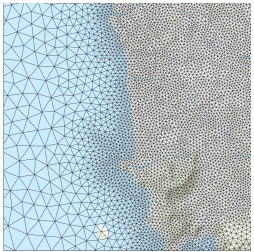


Figure 26: Example of an unstructured mesh



UNSTRUCTURED MESH

Vertices order matter

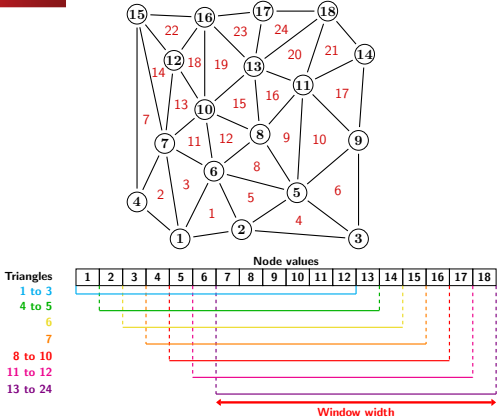


Figure 27: Original mesh

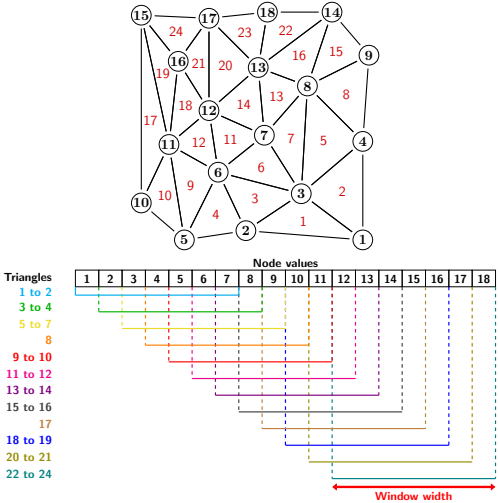


Figure 28: Sorted mesh

UNSTRUCTURED MESH
Reordering mesh with space filling curve

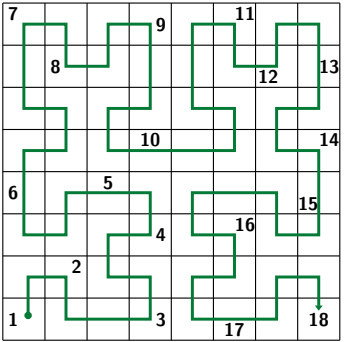


Figure 29: Sorting vertices according to SFC Path

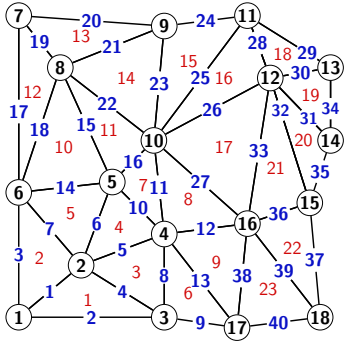


Figure 30: Reordering elements and edges according to vertices order

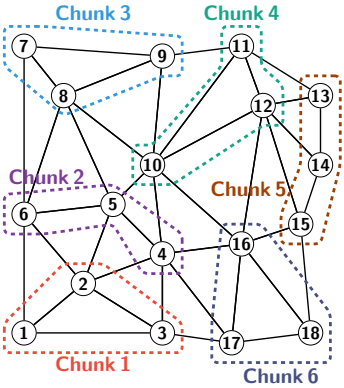


Figure 31: Partitioning of the mesh into chunks of 3 elements.

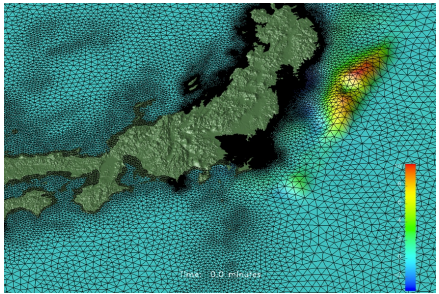
CASE STUDY II: TSUNAMI SIMULATION

Overview

Lexis Project

Alfred-Wegener Institut

German-Indonesian Tsunami Early Warning System



© Alfred-Wegener Institut

Figure 32: TsunAWI computational mesh with an initial condition for the Tōhoku (Japan) Tsunami 2011

CASE STUDY II: TSUNAMI SIMULATION

Compute kernel

```
for(int i = 0; i < nTriangles; ++i)
  grad[i] =
      coef[i][0] * data[topo[i][0]]
    + coef[i][1] * data[topo[i][1]]
    + coef[i][2] * data[topo[i][2]];
```

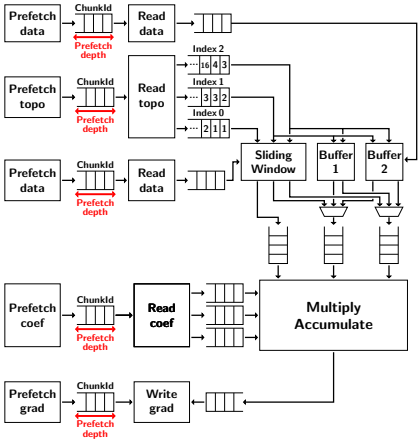


Figure 33: Compute kernel architecture

CASE STUDY II: TSUNAMI SIMULATION
Meshes dataset

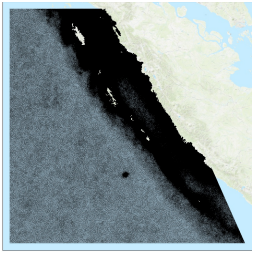


Figure 34: Padang fine-grained mesh topology.

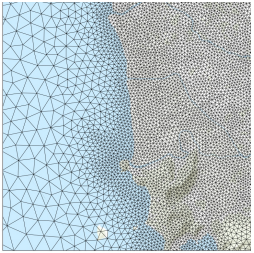


Figure 35: Zoom on coastal area.

Name	# Elements	# Vertices	Size
Padang_C	460 119	231 586	14 Mo
Padang_F	2 470 345	1 242 653	74 Mo
Coquimbo_C	3 396 755	1 709 506	102 Mo
Coquimbo_F	9 762 027	4 887 927	293 Mo
Mediterranean	9 917 645	4 999 404	298 Mo

Tableau 3: Meshes dataset used for tsunami simulation.

CASE STUDY II: TSUNAMI SIMULATION

Performance according to the number of processing elements

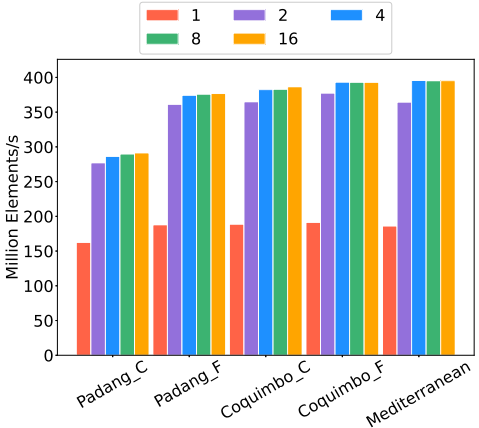


Figure 36: Computation speed in MElements/s according to the number of processing elements.

Analysis

- Large loss of efficiency beyond 2 PEs
- The smaller mesh is processed slower
- Why? Data starvation again!

CASE STUDY II: TSUNAMI SIMULATION

Performance according to the number of processing elements

Explanation

- Local memory bandwidth bottleneck for the two largest meshes
- The smaller the mesh, the less memory bandwidth is used
- Latency is not completely masked for the smallest

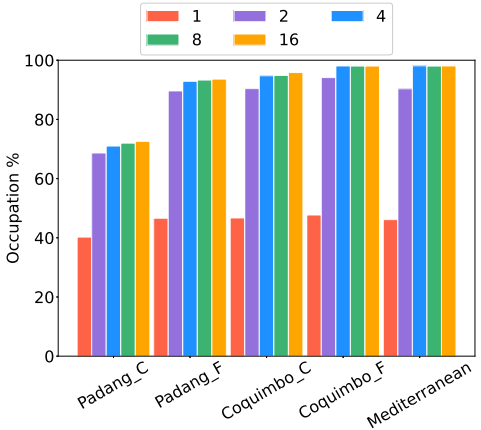


Figure 37: Memory controller activity (occupancy percentage) according to the number of processing elements.

CASE STUDY II: TSUNAMI SIMULATION

Performance according the system topology

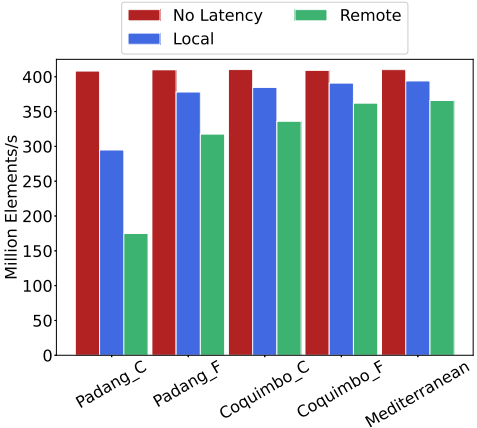


Figure 38: Computation speed in MElements/s according the system topology.

Analysis

- The latency increase slows down processing of all meshes
- The smaller the mesh, the slower the processing

CASE STUDY II: TSUNAMI SIMULATION

Performance according the system topology

Explanation

- The smaller the mesh, the shorter the processing time
- But the loading time of the first chunk remains constant
- So, for smaller meshes, the loading time of the first chunk represents a significant part of the processing time

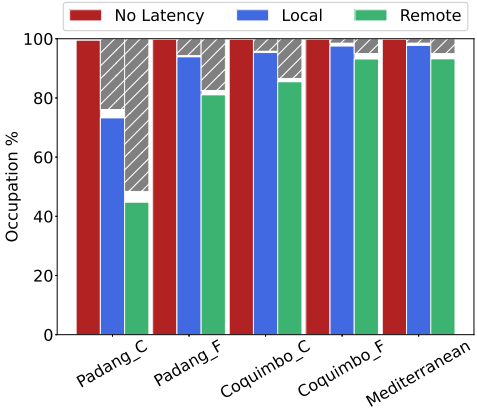


Figure 39: Memory controller activity (occupancy percentage) according the system topology.

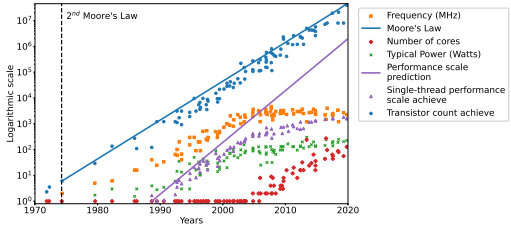
CASE STUDY II: TSUNAMI SIMULATION
Discussion

- Data access latency is impacting at kernel launch
- The shorter the simulation time, the greater the impact
- When reaching the nominal mode of the kernel, latency seems to be masked successfully
- As for the first case study:
  - A data starvation is caused by a local memory bandwidth bottleneck
  - The programming model is well suited for large datasets

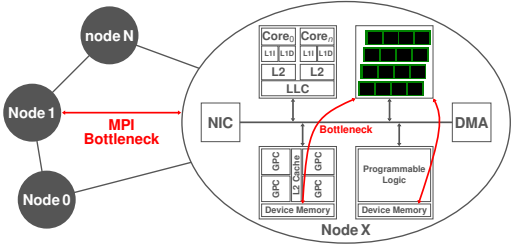
OUTLOOK: 4 - CONCLUSION & OUTLOOK

1. Context & motivation
2. State-of-the-art
3. Contributions & Experimentation
4. Conclusion & Outlook

CONCLUSION
Motivation

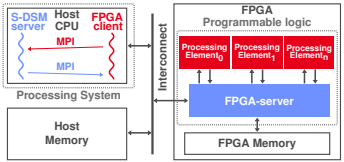


End of Dennard scaling has led to heterogeneous systems



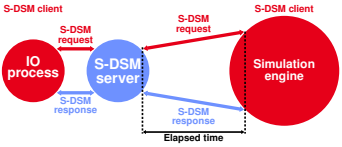
Heterogeneous system programming is complex

CONCLUSION
Contributions



Integrating FPGA into S-DSM

COMPAS'2019



Homemade hybrid simulation tool

RSP'2020

Programming model for irregular compute kernel

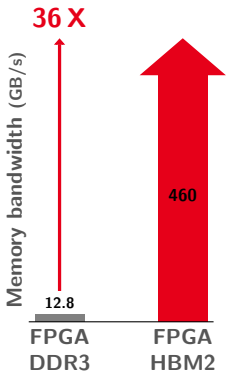
HeteroPar'2021 & COMPAS'2021

Publications:

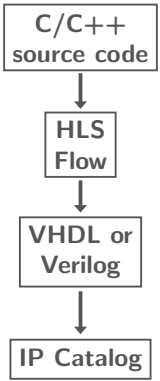
CONCLUSION
Experiment

- The S-DSM acts as a software cache directory for chunks
- Enjoy the same advantages and face the same drawbacks
  - Transparently bring intelligence into data management
  - Prefetch data thanks to chunks partitioning
  - Improve data locality
- The programming model is well suited for large datasets
- Accelerator memory bandwidth remains a bottleneck
- The programming model enables to:
  - Efficiently hides distributed data access latency
  - Almost reach the maximum performance allowed by the accelerator local memory

OUTLOOK



Extend the integration model to FPGAs with HBM technology



Enable High Level Synthesis (HLS) from C/C++ source code

SCIENTIFIC PUBLICATIONS & COMMUNICATIONS

International workshop	Erwan Lenormand et al. "A combined fast/cycle accurate simulation tool for reconfigurable accelerator evaluation: application to distributed data management". In: *International Workshop on Rapid System Prototyping*. 2020
	Erwan Lenormand et al. "Data Management Model to Program Irregular Compute Kernels on FPGA: Application to Heterogeneous Distributed System". In: *Euro-Par 2021: Parallel Processing Workshops*. 2021
National conference	Erwan Lenormand et al. "Unification des mémoires réparties dans un système hétérogène avec accélérateur reconfigurable : exposé de principe". In: *Conférence d'informatique en Parallélisme, Architecture et Système*. 2019
	Erwan Lenormand et al. "Modèle de programmation pour noyaux de calcul irréguliers sur accélérateur reconfigurable dans un système distribué hétérogène". In: *Conférence francophone d'informatique en Parallélisme, Architecture et Système*. 2021
Poster	Erwan Lenormand et al. *Unifying distributed memories in a heterogeneous system with reconfigurable accelerators*. International Summer School ACACES. 2019
Journal	Thierry Goubier et al. "Modeling and Implementing an Earthquake and Tsunami Event-triggered, Time-constrained Impact Assessment Workflow". In: *Frontiers in Earth Science Geohazards and Georisks* (Under review)

Merci!

Commissariat à l'énergie atomique et aux énergies alternatives
Campus Saclay Nano-Innov | F-91191 Gif-sur-Yvette Cedex
www-list.cea.fr
Établissement public à caractère industriel et commercial | RCS Paris B 775 685 019