Ultra-low Power Computing with CGRAs

an architecture, compilation, and application triptych

Kevin J. M. Martin, Philippe Coussy

Univ. Bretagne-Sud
Lab-STICC, UMR CNRS 6285
Lorient, France

20/06/2022
Workshop on Reconfigurable Computing

Outline

Introduction

Mismatching energy constraints and trends

Energy constraints

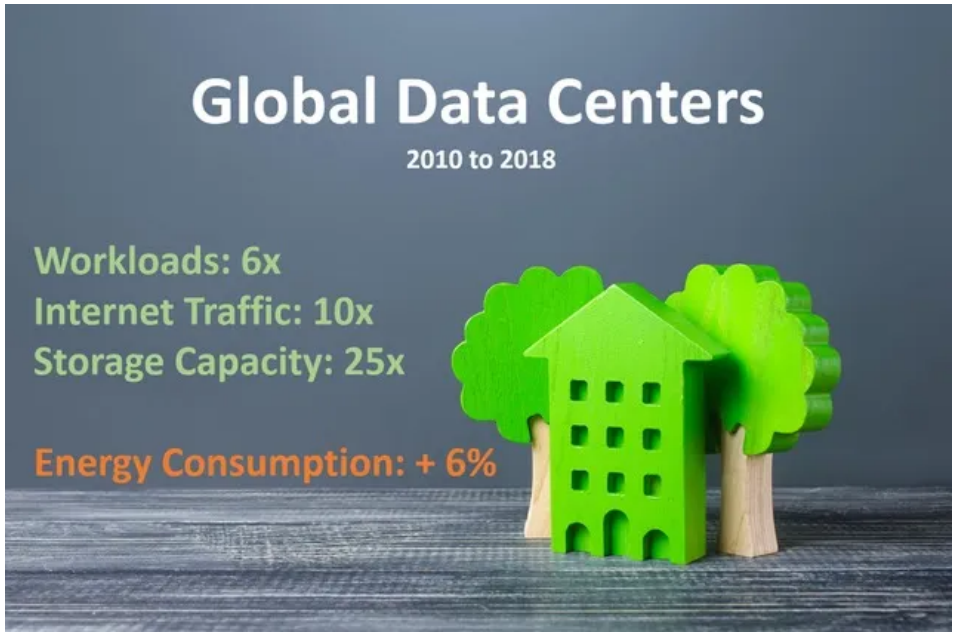
- Carbon neutral society
  - Low carbon footprint
- Battery powered devices

Energy trends

- Consume everywhere
  - Internet of (every-)Things (IoT), IoET, Industrial IoT (IIoT), fog/edge computing
- Energy hungry workloads
  - AI/ML, autonomous driving, computer vision, video streaming, etc.
- The scary figures
  - Communication technology is expected to become 21% of global electricity usage [2] (903 TWh)
  - the “Global DataSphere” will grow to 175 ZB by 2025 [51]
  - 62.7% of the total system energy, on average, is spent on data movement between main memory and the compute units [8]

Introduction

Issues and challenges



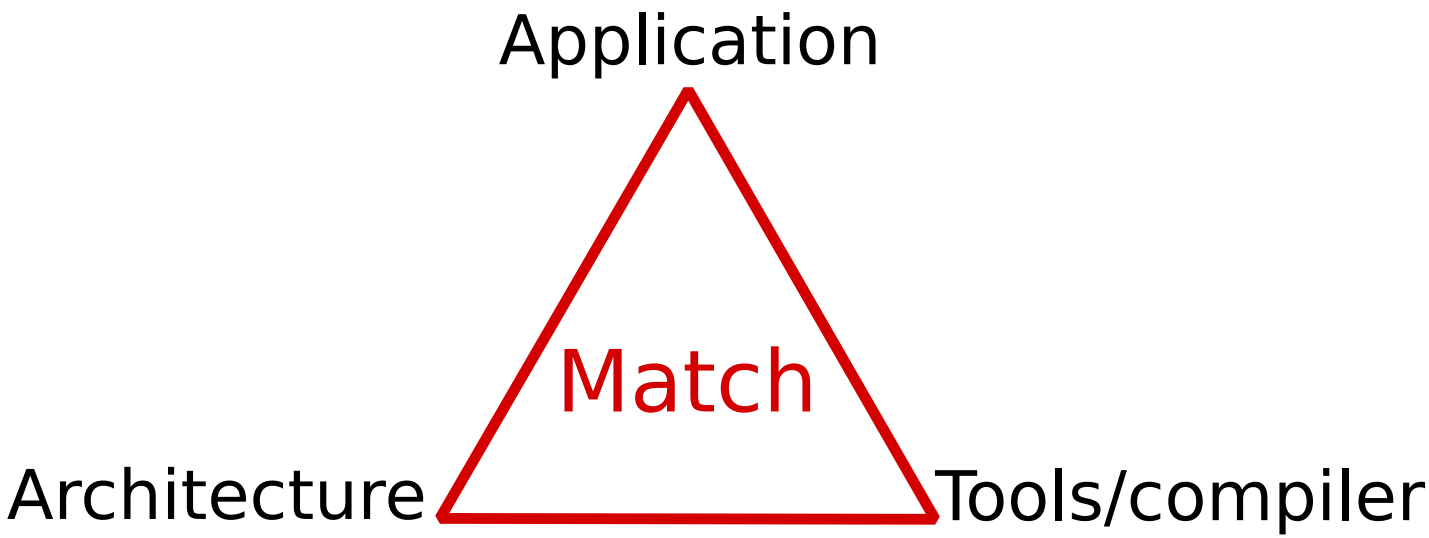
Source: https://i2.wp.com/semiengineering.com

Ultra-low-power

Energy efficient architectures for the IoT-edge-cloud computing continuum

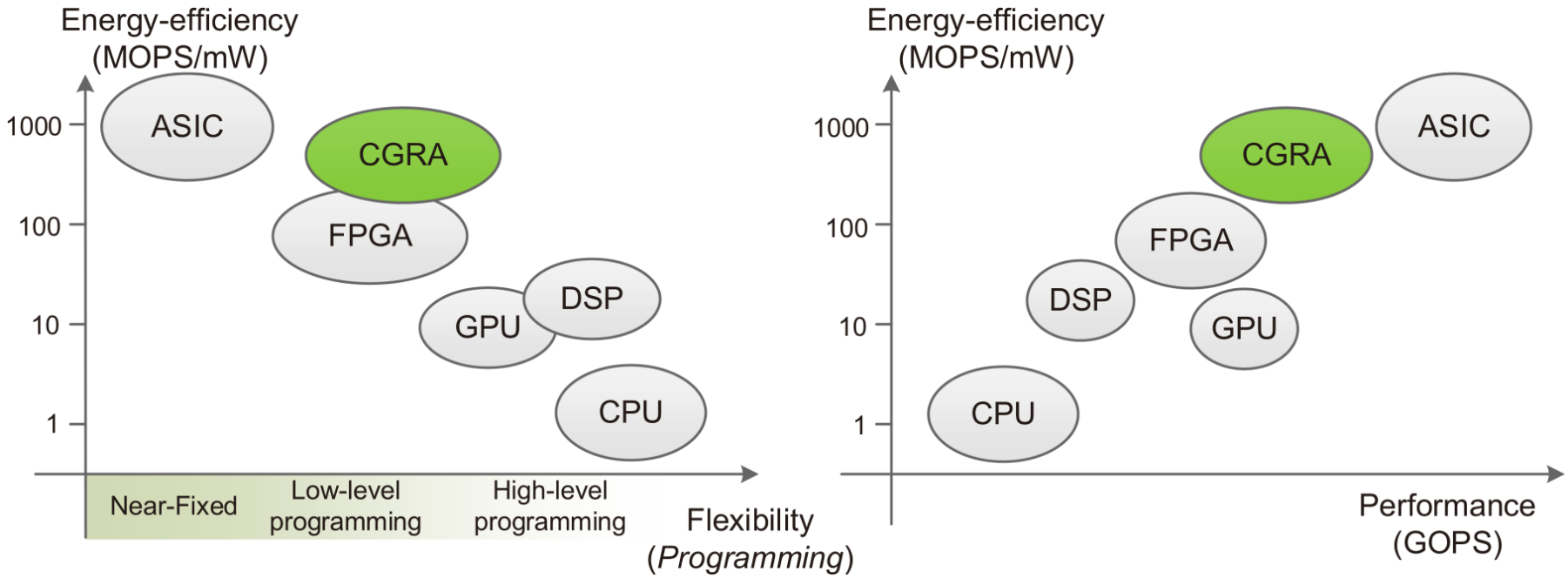
Introduction

The quest for (energy) performance



Introduction

Architectural solutions



Source: Liu et al. [39]

Coarse Grained Reconfigurable Architecture

- energy efficient solution
- trade-off performance/flexibility

Three decades of CGRA

CGRA architectures

30 years of architectures

Inventory of existing CGRAs done in [29, 56, 13, 60, 39, 48]

- Hartenstein 2001 [29]: the first decade
- De Sutter et al. 2010 [19]: book chapter
- Wijtvliet et al. 2016 [60]: 25 years of CGRAs
- Liu et al. 2019 [39]: taxonomy, classification
- Podebas et al. 2020 [48]: performance perspective

Main limitations

- Many programmed by hand (assembly level)
- Unadapted programming model
- How to efficiently make use of the available computing resources?

Three decades of CGRA

Example of CGRA

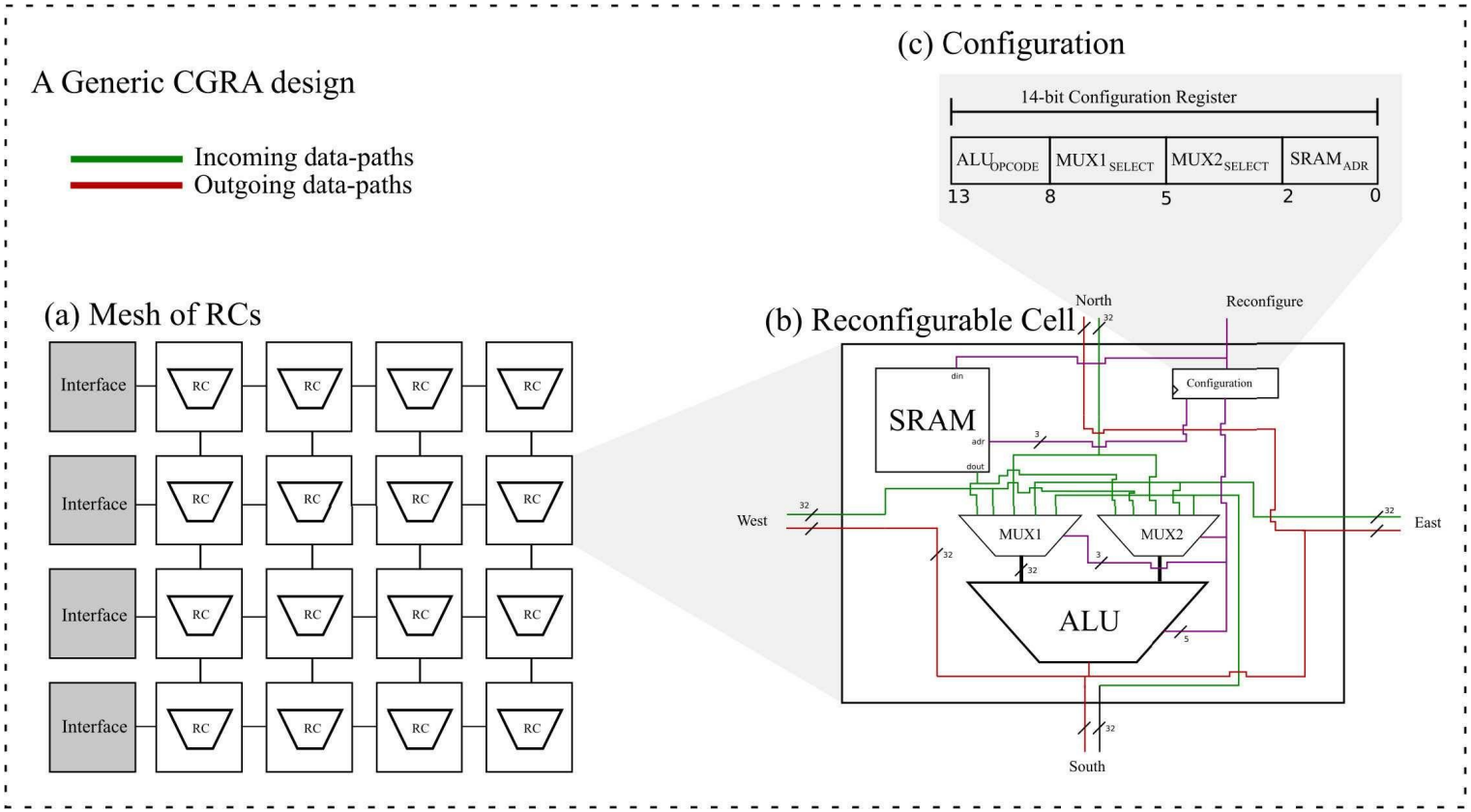


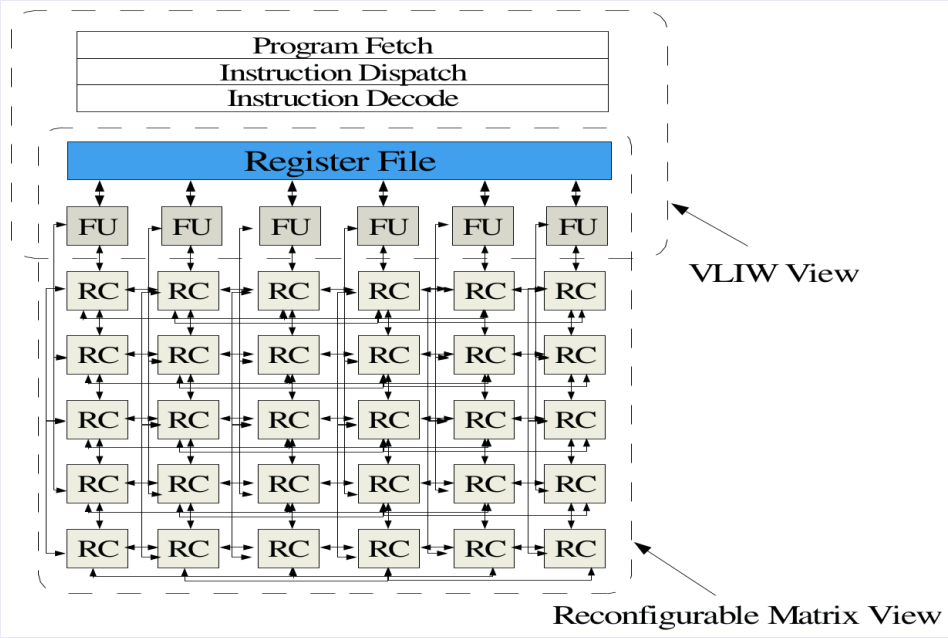
Illustration of a simple CGRA taken from [48], showing the mesh topology (a), the internal architecture of the Reconfigurable Cell, RC (b), and an example of the configuration register (c).

[48] A. Podobas, K. Sano, and S. Matsuoka. A survey on coarse-grained reconfigurable architectures from a performance perspective. *IEEE Access*, 2020.

Three decades of CGRA

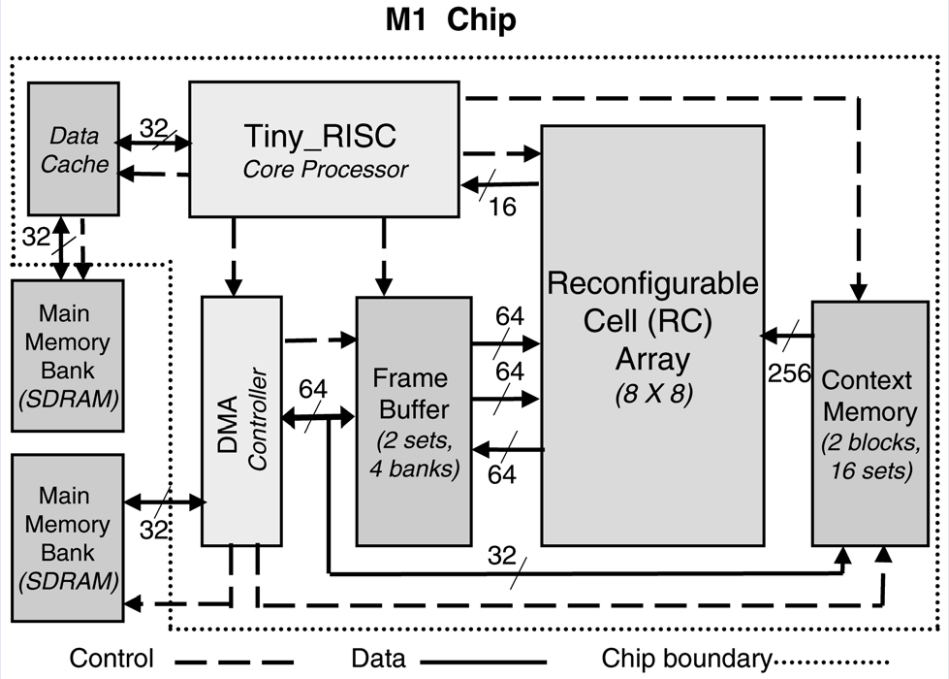
Coupling with a host processor

Tightly coupled



ADRES [42]

Loosely coupled



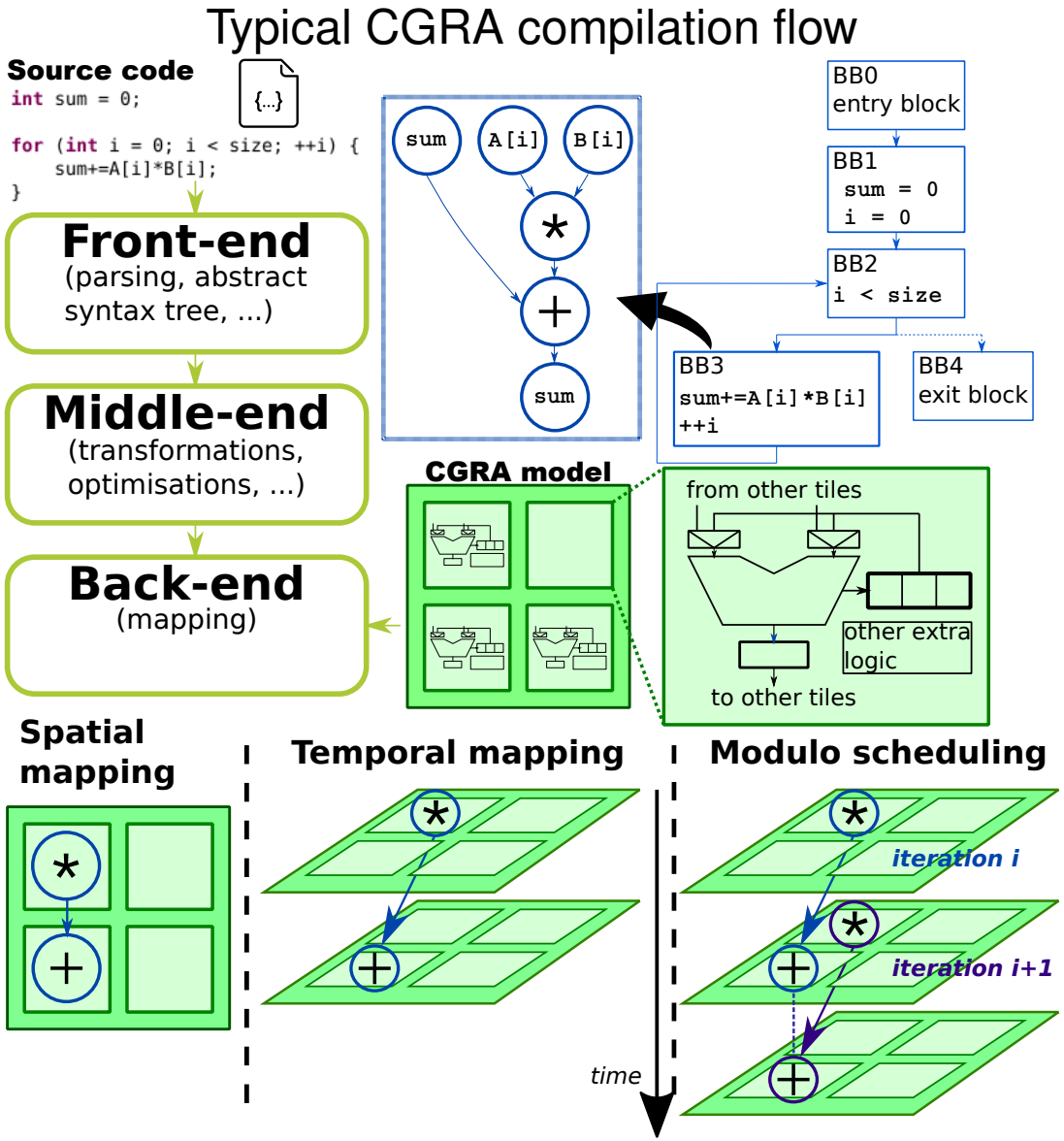
MorphoSys [53]

[42] Mei et al. ADRES: An architecture with tightly coupled vliw processor and coarse-grained reconfigurable matrix. *FPL*, 2003

[53] Singh et al. Morphosys: an integrated reconfigurable system for data-parallel and computation-intensive applications. *IEEE Transactions on Computers*, 2000.

Three decades of CGRA

CGRA mapping



Three decades of CGRA

CGRA mapping

CGRA mapping problem formulation

Bind in place and schedule in time operations of the application on the cells of the CGRA, in a short time, while guaranteeing the dependencies, such that the application executes as fast as possible

CGRA mapping problem

- Solve 2 NP-complete problems: scheduling and binding
  - + register allocation
- Mapping might fail!

	Approximate methods			Exact methods	
	Heuristics	Meta-heuristics		ILP/B&B	CSP
		Population-based	local search		
Spatial mapping	[30, 62, 37]	GA [34]	SA [58, 23]	ILP [12, 45, 62]	
Temporal mapping	[59, 7, 36, 65, 46, 18, 24, 10]		SA [40]	ILP [9] B&B [32]	CP [49] SAT [43] SMT [20]
Binding	[61, 31, 27, 47, 17, 26]	QEA [35]	SA [21, 30, 52]	ILP [25, 35]	
Scheduling	[17, 26, 35, 27, 6, 65, 52, 5]			ILP [25, 44]	

A review of binding and scheduling techniques for automated spatial and temporal mapping of applications on CGRAs.

Three decades of CGRA

CGRA mapping

Mapping if-then-else (ITE) constructs

1. Full predication [3]
2. Partial predication [11]
3. Dual-issue single execution [28, 33, 63]
4. Direct CDFG mapping [14]

Three decades of CGRA

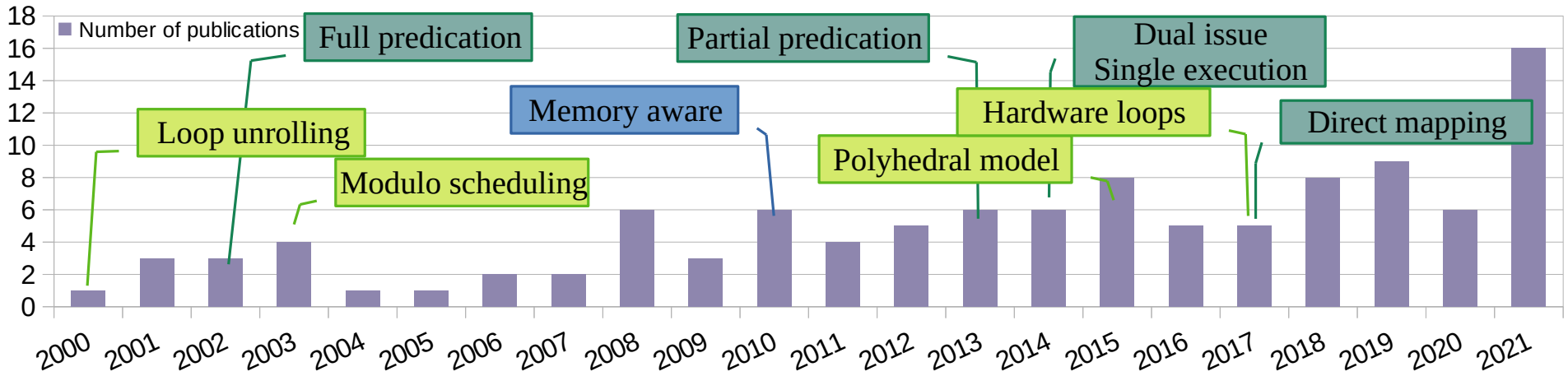
CGRA mapping

Mapping loops

1. Modulo scheduling
   - Modulo routing resource graph (MRRG) [41, 33]
   - Graph-based approaches [18, 46]
   - ILP
2. Direct CDFG mapping [14]
3. Hardware loops [4, 54, 57]

Three decades of CGRA

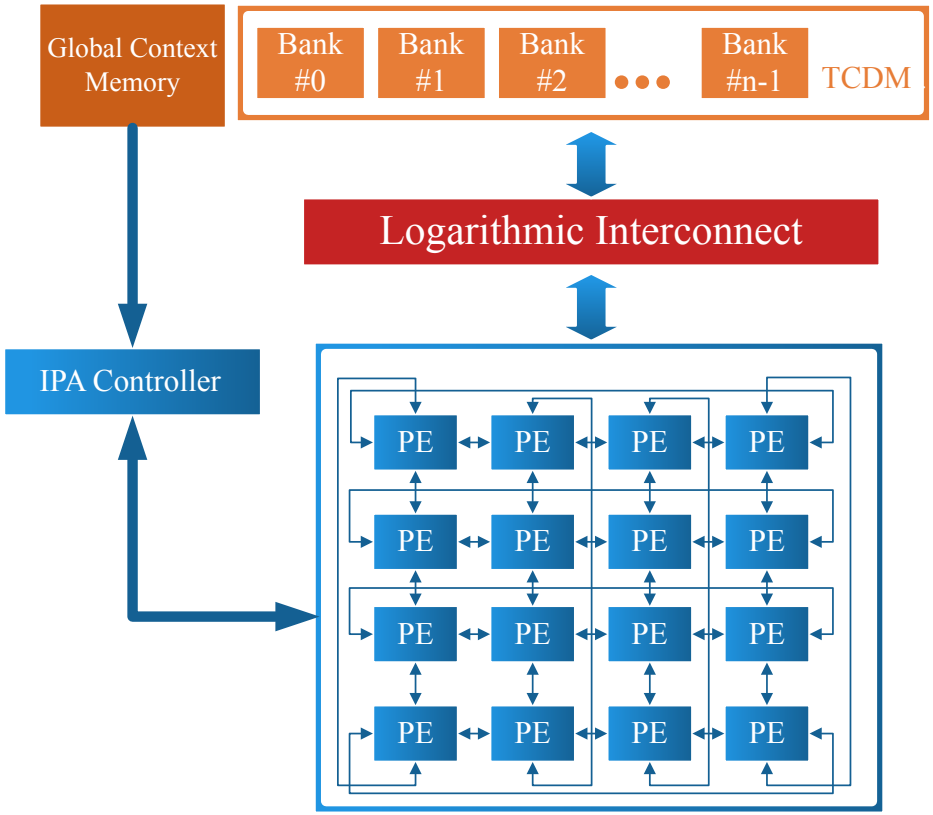
Timeline



Integrated Programmable-Array (IPA)

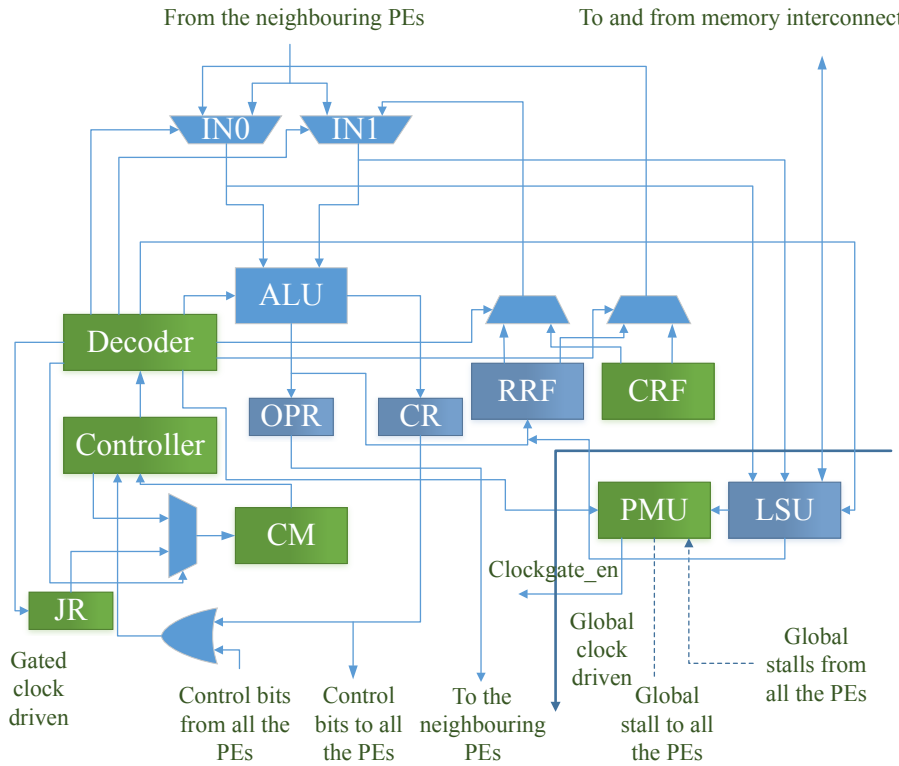
Integrated Programmable Array (IPA)

Integration with data on-chip memory



TCDM: Tightly Coupled Data Memory
PE: Processing Element

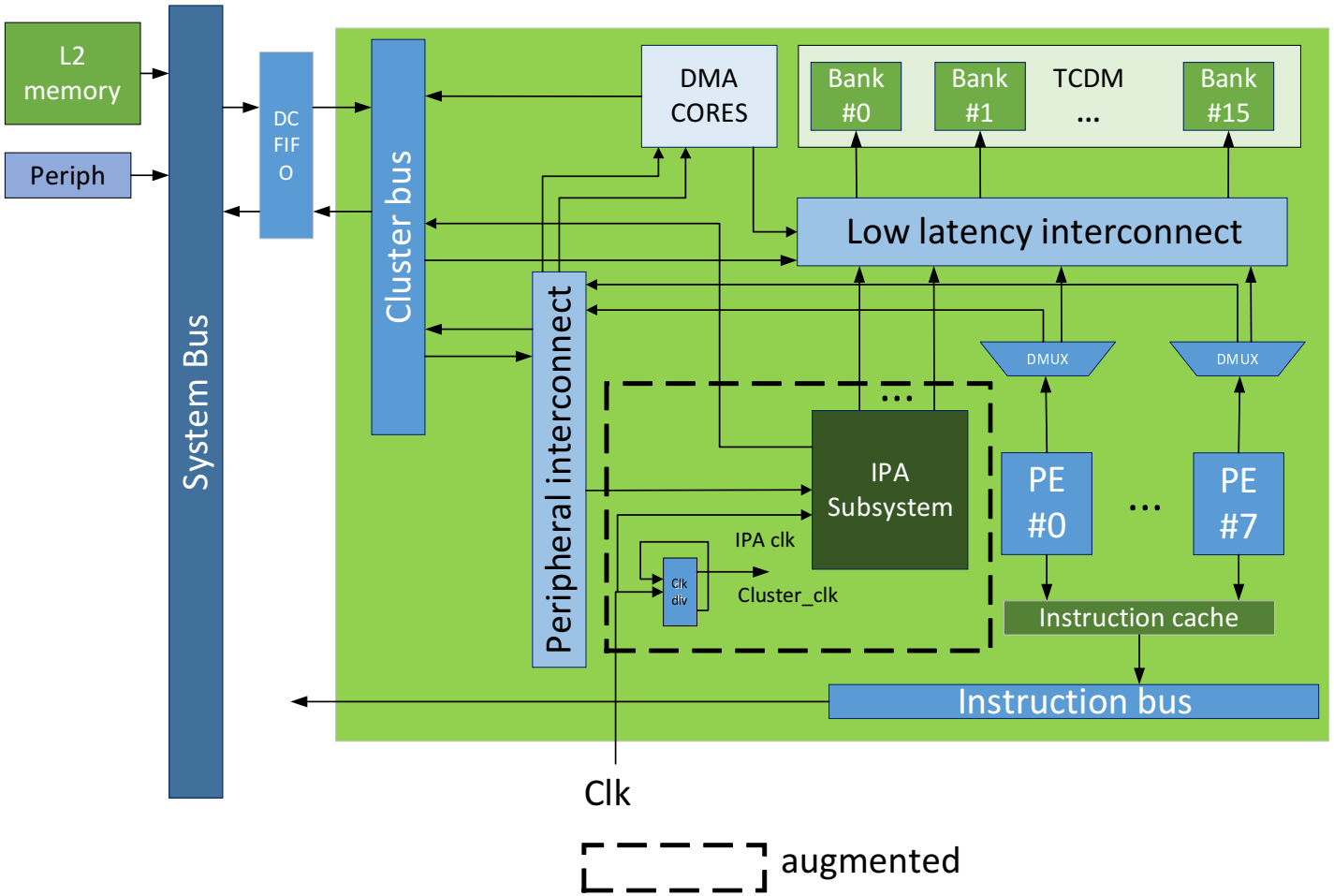
Inside a Processing Element



PMU: Power Management Unit
RRF: *Regular* Register file
CRF: Constant Register File
CM: Context Memory
OPR: Output Register
CR: Control Register

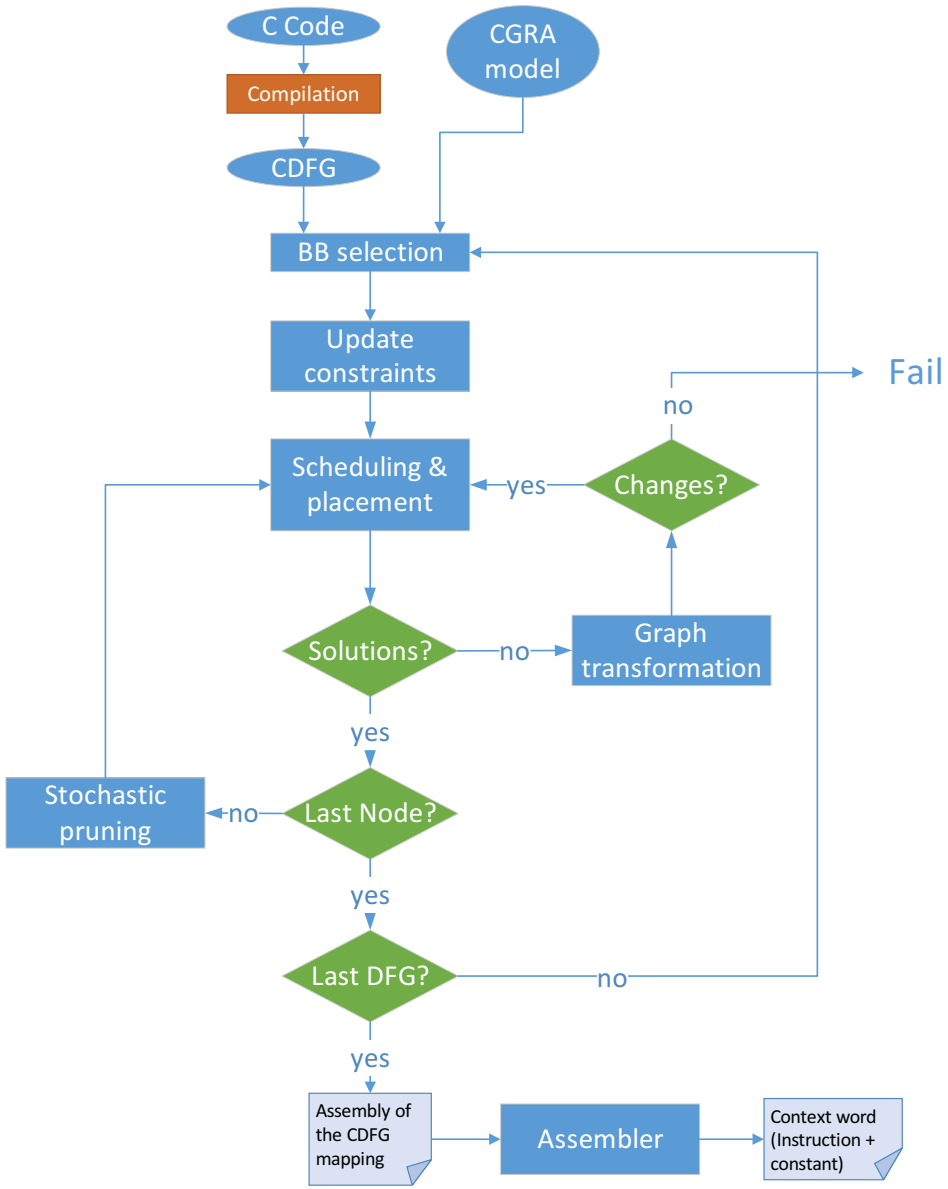
Integrated Programmable-Array (IPA)

Integration in a PULP cluster



Integrated Programmable-Array (IPA)

Compilation flow



Compilation method

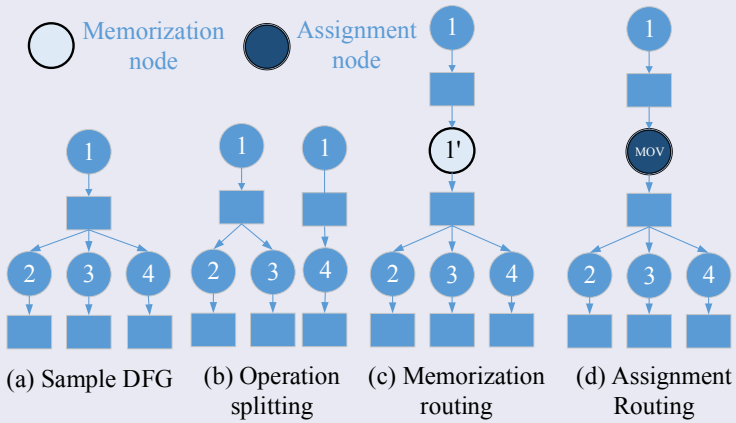
- Except array inputs and outputs, all the scalar variables are placed in the register files of the PEs
- Modified Forward traversal of CDFG to minimize the number of constraints
- Introduce routing to reach the correct variable, while mapping the basic blocks

Compilation steps

DFG mapping

- Backward traversal of operation nodes
- List scheduling based on priority (fanout)
- Partial Levi's algorithm to find common subgraphs
- Graph transformations
- Stochastic pruning
  - Keep a minimum number of partial mappings to find a complete mapping
  - Keep a maximum number of partial mappings to manage scalability

Graph transformations



Integrated Programmable-Array (IPA)

Experimental Setup

Technology and tools

- STMicroelectronics 28 nm UTBB FD-SOI
- Synopsys design compiler 2014.09-SP4
- Synopsys PrimePower 2013.12-SP3
- Questa Sim-64 10.5c

CGRA

- 4×4 array with 16 PEs
- 20×32-bit (Instructions), a 32×8-bit RRF, and 32×16-bit CRF
- 100 MHz
- clock gating

Integrated Programmable-Array (IPA)

Results

Overall instructions executed and energy consumption in IPA vs CPU [16]

Kernels		FIR	MatM (16×16)	Convolution	SepFilter	NonSepFilter	FFT	DC Filter
IPA	**Configuration cycles**	71	88	88	90	98	87	145
	Execution cycles	6 071	11 940	56 241	827 685	1 852 382	8 076	4 748
	Total number of instructions executed	44 294	110 946	531 815	7 349 843	17 486 486	76 310	28 868
	Active PEs/cycle (%)	46.1	58.5	59.2	55.5	59	59.7	39.5
	Energy (μJ)	0.022	0.043	0.202	2.98	6.669	0.032	0.017
	Energy (μJ) in non-clock-gated IPA	0.047	0.077	0.479	7.152	11.704	0.063	0.045
CPU	**Execution cycles**	37 677	96 256	616 805	5 982 730	9 084 101	164 480	50 085
	Energy (μJ)	0.132	0.337	2.159	20.94	31.794	0.576	0.175
	Speed-up	6.21x	8.06x	10.97x	7.23x	**4.9x**	**20.3x**	10.55x
	Energy-gain	6x	7.84x	10.69x	7.03x	**4.77x**	**18x**	10.29x

[16] Satyajit Das et al. *An energy-efficient integrated programmable array accelerator and compilation flow for near-sensor ultralow power processing*, IEEE TCAD, 2019.

Integrated Programmable-Array (IPA)

Results

Performance evaluation in execution time (ns) for different configurations in the heterogeneous platform [15]

Kernels	Data size (KB)	Single-core (ns)	Multi-core (ns)	Speed-up in multi-core (x)	IPA (ns)	Speed-up in IPA (x)
MatMul	8	3,358,740	435,180	7.72	432,630	7.76
Conv	8	9,733,380	1,520,840	6.4	1,494,860	6.51
FFT	1	767,640	142,720	5.38	94,510	8.12
FIR	0.84	182,500	33,460	5.45	33,410	5.46
Sep Filter	10	39,870,420	6,404,160	6.23	6,334,700	6.29
Sobel Filter	10	117,024,880	40,894,260	2.86	28,865,890	4.05
GCD	0.01	2,951,160	2,951,160	1	61,1300	4.83
Cordic	0.06	9,000	7,000	1.29	3,610	2.49
Manh Dist	8	244,640	164,640	1.49	70,300	3.48

[15] Satyajit Das et al. *A heterogeneous cluster with reconfigurable accelerator for energy efficient near-sensor data analytics*, ISCAS, 2018.

Integrated Programmable-Array (IPA)

Results

Energy consumption evaluation in μ J for different configurations in the heterogeneous platform [15]

Kernels	Single-core	Multi-core	IPA	
			Energy	of Active PEs/cycle
MatMul	1.247	0.313	0.208	58.5
Convolution	2.876	1.095	0.658	59.2
FFT	0.292	0.087	0.042	59.7
FIR	0.08	0.026	0.026	46.1
Separable filter	16.663	4.611	4.28	55.5
Sobel Filter	51.491	29.444	12.701	51.2
GCD	1.151	1.151	0.257	6.25
Cordic	0.004	0.003	0.001	50
ManhDistance	0.1	0.095	0.03	48.5

[15] Satyajit Das et al. *A heterogeneous cluster with reconfigurable accelerator for energy efficient near-sensor data analytics*, ISCAS, 2018.

Integrated Programmable-Array (IPA)

Comparison with state of the art architectures

	Ultra-low-power CGRAs			High-performance CGRAs			
	ULP-SRP [34]	**CMA [55]**	**IPA [17]**	**HyCube [33]**	**Revel [75]**	**SGMF [71]**	**SNAFU**
Fabric size	3×3	8×10	4×4	4×4	5×5	8×8 + 32 mem	N×N (6×6 in SNAFU-ARCH)
NoC	Neighbors only	Neighbors only	Neighbors only	Static, bufferless, multi-hop	Static & dynamic NoCs (2×)	Dynamic routing	Static, bufferless, multi-hop
PE assignment	Static	Static	Static	Static	Static *or* dynamic	Dynamic	Static
Time-share PEs?	Yes	Yes	Yes	Yes	Yes	Yes	No
PE firing	Static	Static	Static	Static	Static *or* dynamic	Dynamic	Dynamic
Heterogeneous PEs?	No	No	No	No	Yes	Yes	Yes
Buffering (approx.)	—	—	188 B / PE	272 B / PE	≈1 KB / PE	≫1 KB / PE	40 B / PE
Power	22 mW	11 mW	3–5 mW	15–70 mW	160 mW	20 W	<1 mW
MOPS/mW (approx.)	30–100	100–200	140	60–90	60	60	305

Table I: Architectural comparison of SNAFU to several prior CGRAs.

Taken from [23]

[23] Gobieski et al. Snafu: An ultra-low-power, energy-minimal cgra-generation framework and architecture. *ISCA*, 2021

Integrated Programmable-Array (IPA)

IPA features

- Architectural support
  - PMU: Power Management Unit
  - CRF: Constant Register File
  - Global synchronization
  - ISA-based configurations (20/21-bits)
- Compiler
  - Direct CDFG mapping (register allocation approach)
  - Backward traversal of operation nodes
  - List scheduling based on priority (fanout)
  - Partial Levi's algorithm to find common subgraphs
  - Dynamic Graph transformations
  - Stochastic pruning

Summary

IPA: Integrated Programmable Array

- Ultra-low power domain
- Integer operations
- CDFG support

TRANSPIRE: transprecision and SIMD support

- Floating-point operations
- Multi-cycle operators

Perspectives

CGRA

- Still ever promising approach?
- New momentum
  - Xilinx AI engine
  - Reconfigurable Dataflow Architecture/Accelerator (RDA)

Embedded systems

- Small grids (4x4, 5x5, 6x6)
- Temporal mapping
- (Ultra) Low Power

HPC systems

- (Very) Large grids (100s to 1000s of cells)
- Spatial mapping
- HPC: High Power Consuming

Perspectives

CGRAs for AI

Artificial Intelligence (AI) applications

- Xilinx AI-engine [22]
- Reconfigurable Dataflow Architecture [50]
- Reconfigurable Dataflow Accelerator [64]

Scalability challenge

AI for CGRAs

- Machine learning to optimize the mapping [38]
- Single model: functional + non functional contraints

CGRA + emerging memory technologies and organisations

- MRAM, FeRAM, ReRAM, PCM, ...
- Computing in/near memory

Open-source frameworks

- DSAGEN [58]
- OpenCGRA [55]
- CGRA-ME [1]
- CCF

THANK YOU

... and thanks to Philippe, Matthieu, Gwenolé, Luca, Davide, Mickaël, Thomas, Satyajit, Rohit, Chilanka

References I

[1] Jason Anderson, Rami Beidas, Vimal Chacko, Hsuan Hsiao, Xiaoyi Ling, Omar Ragheb, Xinyuan Wang, and Tianyi Yu.
Cgra-me: An open-source framework for cgra architecture and cad research : (invited paper).
In *2021 IEEE 32nd International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pages 156–162, 2021.

[2] Anders S. G. Andrae and Tomas Edler.
On global electricity usage of communication technology: Trends to 2030.
Challenges, 6(1):117–157, 2015.

[3] M. L. Anido, A. Paar, and N. Bagherzadeh.
Improving the operation autonomy of simd processing elements by using guarded instructions and pseudo branches.
In *Proceedings Euromicro Symposium on Digital System Design. Architectures, Methods and Tools*, pages 148–155, 2002.

[4] Mahesh Balasubramanian, Shail Dave, Aviral Shrivastava, and Reiley Jeyapaul.
Laser: A hardware/software approach to accelerate complicated loops on cgras.
In *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1069–1074, 2018.

[5] Mahesh Balasubramanian and Aviral Shrivastava.
Crimson: Compute-intensive loop acceleration by randomized iterative modulo scheduling and optimized mapping on cgras.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 39(11):3300–3310, 2020.

[6] Nikhil Bansal, Sumit Gupta, Nikil Dutt, and Alexandru Nicolau.
Analysis of the performance of coarse-grain reconfigurable architectures with different processing element configurations.
In *Workshop on Application Specific Processors, held in conjunction with the International Symposium on Microarchitecture (MICRO), 2003*, 2003.

[7] Kiran Bondalapati and Viktor K. Prasanna.
Mapping loops onto reconfigurable architectures.
In Reiner W. Hartenstein and Andres Keevallik, editors, *Field-Programmable Logic and Applications From FPGAs to Computing Paradigm*, pages 268–277, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.

[8] Amirali Boroumand, Saugata Ghose, Youngsok Kim, Rachata Ausavarungnirun, Eric Shiu, Rahul Thakur, Daehyun Kim, Aki Kuusela, Allan Knies, Parthasarathy Ranganathan, and Onur Mutlu.
Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks.
In *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '18, pages 316–331, New York, NY, USA, 2018. ACM.
event-place: Williamsburg, VA, USA.

[9] Janina A Brenner, JC Van Der Veen, Sándor P Fekete, J Oliveira Filho, and Wolfgang Rosenstiel.
Optimal Simultaneous Scheduling, Binding and Routing for Processor-Like Reconfigurable Architectures.
In *FPL*, pages 1–6, August 2006.

References II

[10] Michael Canesche, Marcelo Menezes, Westerley Carvalho, Frank Sill Torres, Peter Jamieson, José Augusto Nacif, and Ricardo Ferreira.
Traversal: A fast and adaptive graph-based placement and routing for cgras.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 40(8):1600–1612, 2021.

[11] Kyungwook Chang and K. Choi.
Mapping control intensive kernels onto coarse-grained reconfigurable array architecture.
In *2008 International SoC Design Conference*, volume 01, pages I–362–I–365, Nov 2008.

[12] S. Alexander Chin and Jason H. Anderson.
An architecture-agnostic integer linear programming approach to cgra mapping.
In *Proceedings of the 55th Annual Design Automation Conference*, DAC '18, New York, NY, USA, 2018. Association for Computing Machinery.

[13] Kiyoung Choi.
Coarse-grained reconfigurable array: Architecture and application mapping.
IPSJ Transactions on System LSI Design Methodology, 4:31–46, 2011.

[14] S. Das, K. J. M. Martin, P. Coussy, D. Rossi, and L. Benini.
Efficient mapping of CDFG onto coarse-grained reconfigurable array architectures.
In *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 127–132, Jan 2017.

[15] Satyajit Das, Kevin J. M. Martin, Philippe Coussy, and Davide Rossi.
A heterogeneous cluster with reconfigurable accelerator for energy efficient near-sensor data analytics.
In *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5, 2018.

[16] Satyajit Das, Kevin J. M. Martin, Davide Rossi, Philippe Coussy, and Luca Benini.
An energy-efficient integrated programmable array accelerator and compilation flow for near-sensor ultralow power processing.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(6):1095–1108, 2019.

[17] Satyajit Das, Thomas Peyret, Kevin J. M. Martin, Gwenolé Corre, Mathieu Thevenin, and Philippe Coussy.
A scalable design approach to efficiently map applications on cgras.
In *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 655–660, July 2016.

[18] Shail Dave, Mahesh Balasubramanian, and Aviral Shrivastava.
Ramp: Resource-aware mapping for cgras.
In *Proceedings of the 55th Annual Design Automation Conference*, DAC '18, New York, NY, USA, 2018. Association for Computing Machinery.

[19] Bjorn De Sutter, Praveen Raghavan, and Andy Lambrechts.
Coarse-Grained Reconfigurable Array Architectures, pages 449–484.
Springer US, Boston, MA, 2010.

References III

[20] Caleb Donovick, Makai Mann, Clark Barrett, and Pat Hanrahan.
Agile smt-based mapping for cgras with restricted routing networks.
In *2019 International Conference on ReConFigurable Computing and FPGAs (ReConFig)*, pages 1–8, 2019.

[21] Stephen Friedman, Allan Carroll, Brian Van Essen, Benjamin Ylvisaker, Carl Ebeling, and Scott Hauck.
SPR: An Architecture-adaptive CGRA Mapping Tool.
In *FPGA*, pages 191–200. ACM, 2009.

[22] Alok G.
Architecture apocalypse dream architecture for deep learning inference and compute-versal ai core.
In *Embedded World*, 2020.

[23] Graham Gobieski, Ahmet Oguz Atli, Kenneth Mai, Brandon Lucia, and Nathan Beckmann.
Snafu: An ultra-low-power, energy-minimal cgra-generation framework and architecture.
In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pages 1027–1040, 2021.

[24] Jiangyuan Gu, Shouyi Yin, Leibo liu, and Shaojun Wei.
Stress-aware loops mapping on cgras with dynamic multi-map reconfiguration.
IEEE Transactions on Parallel and Distributed Systems, 29(9):2105–2120, 2018.

[25] Yijiang Guo, Jiarui Wang, Jiaxi Zhang, and Guojie Luo.
Formulating data-arrival synchronizers in integer linear programming for cgra mapping.
In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 943–948, 2021.

[26] Mahdi Hamzeh, Aviral Shrivastava, and Sarma Vrudhula.
EPIMap: using epimorphism to map applications on CGRAs.
In *DAC*, pages 1284–1291. ACM, 2012.

[27] Mahdi Hamzeh, Aviral Shrivastava, and Sarma Vrudhula.
REGIMap: register-aware application mapping on coarse-grained reconfigurable architectures (CGRAs).
In *DAC*, pages 18:1–18:10. ACM, 2013.

[28] Mahdi Hamzeh, Aviral Shrivastava, and Sarma Vrudhula.
Branch-aware loop mapping on cgras.
In *Proceedings of the 51st Annual Design Automation Conference*, DAC '14, page 1–6, New York, NY, USA, 2014. Association for Computing Machinery.

[29] R. Hartenstein.
A decade of reconfigurable computing: A visionary retrospective.
In *Proceedings of the Conference on Design, Automation and Test in Europe*, DATE '01, page 642–649. IEEE Press, 2001.

References IV

[30] Akira Hatanaka and Nader Bagherzadeh.
A modulo scheduling algorithm for a coarse-grain reconfigurable array template.
In *2007 IEEE International Parallel and Distributed Processing Symposium*, pages 1–8, 2007.

[31] Jong-eun Lee, Kiyoung Choi, and N. D. Dutt.
Compilation approach for coarse-grained reconfigurable architectures.
IEEE Design Test of Computers, 20(1):26–33, 2003.

[32] Manupa Karunaratne, Cheng Tan, Aditi Kulkarni, Tulika Mitra, and Li-Shiuan Peh.
Dnestmap: Mapping deeply-nested loops on ultra-low power cgras.
In *Proceedings of the 55th Annual Design Automation Conference*, DAC '18, New York, NY, USA, 2018. Association for Computing Machinery.

[33] Manupa Karunaratne, Dhananjaya Wijerathne, Tulika Mitra, and Li-Shiuan Peh.
4d-cgra: Introducing branch dimension to spatio-temporal application mapping on cgras.
In *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8, 2019.

[34] Takuya Kojima, Nguyen Anh Vu Doan, and Hideharu Amano.
Genmap: A genetic algorithmic approach for optimizing spatial mapping of coarse-grained reconfigurable architectures.
IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 28(11):2383–2396, 2020.

[35] Ganghee Lee, Kiyoung Choi, and N.D. Dutt.
Mapping Multi-Domain Applications Onto Coarse-Grained Reconfigurable Architectures.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 30(5):637–650, May 2011.

[36] Jinho Lee and Trevor E. Carlson.
Ultra-fast cgra scheduling to enable run time, programmable cgras.
In *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pages 1207–1212, 2021.

[37] Zhaoying Li, Dhananjaya Wijerathne, Xianzhang Chen, Anuj Pathania, and Tulika Mitra.
Chordmap: Automated mapping of streaming applications onto cgra.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pages 1–1, 2021.

[38] Dajiang Liu, Shouyi Yin, Guojie Luo, Jiaxing Shang, Leibo Liu, Shaojun Wei, Yong Feng, and Shangbo Zhou.
Data-flow graph mapping optimization for cgra with deep reinforcement learning.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 38(12):2271–2283, 2019.

[39] Leibo Liu, Jianfeng Zhu, Zhaoshi Li, Yanan Lu, Yangdong Deng, Jie Han, Shouyi Yin, and Shaojun Wei.
A survey of coarse-grained reconfigurable architecture and design: Taxonomy, challenges, and applications.
ACM Comput. Surv., 52(6), October 2019.

References V

[40] Bingfeng Mei, S. Vernalde, D. Verkest, H. De Man, and R. Lauwereins.
DRESC: a retargetable compiler for coarse-grained reconfigurable architectures.
In *FPT*, pages 166–173, December 2002.

[41] Bingfeng Mei, S. Vernalde, D. Verkest, H. De Man, and R. Lauwereins.
Exploiting loop-level parallelism on coarse-grained reconfigurable architectures using modulo scheduling.
In *2003 Design, Automation and Test in Europe Conference and Exhibition*, pages 296–301, 2003.

[42] Bingfeng Mei, Serge Vernalde, Diederik Verkest, Hugo De Man, and Rudy Lauwereins.
Adres: An architecture with tightly coupled vliw processor and coarse-grained reconfigurable matrix.
In Peter Y. K. Cheung and George A. Constantinides, editors, *Field Programmable Logic and Application*, pages 61–70, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[43] Yukio Miyasaka, Masahiro Fujita, Alan Mishchenko, and John Wawrzynek.
Sat-based mapping of data-flow graphs onto coarse-grained reconfigurable arrays.
In Andrea Calimera, Pierre-Emmanuel Gaillardon, Kunal Korgaonkar, Shahar Kvatinsky, and Ricardo Reis, editors, *VLSI-SoC: Design Trends*, pages 113–131, Cham, 2021. Springer International Publishing.

[44] Song Mu, Yi Zeng, and Bo Wang.
Routability-enhanced scheduling for application mapping on cgras.
IEEE Access, 9:92358–92366, 2021.

[45] Tony Nowatzki, Michael Sartin-Tarm, Lorenzo De Carli, Karthikeyan Sankaralingam, Cristian Estan, and Behnam Robatmili.
A general constraint-centric scheduling framework for spatial architectures.
SIGPLAN Not., 48(6):495–506, jun 2013.

[46] Hyunchul Park, Kevin Fan, Scott A. Mahlke, Taewook Oh, Heeseok Kim, and Hong-seok Kim.
Edge-centric Modulo Scheduling for Coarse-grained Reconfigurable Architectures.
In *PACT*. ACM, 2008.

[47] Thomas Peyret, Gwenolé Corre, Mathieu Thevenin, Kevin J. M. Martin, and Philippe Coussy.
Efficient application mapping on CGRAs based on backward simultaneous scheduling/binding and dynamic graph transformations.
In *ASAP*, pages 169–172, June 2014.

[48] A. Podobas, K. Sano, and S. Matsuoka.
A survey on coarse-grained reconfigurable architectures from a performance perspective.
IEEE Access, 8:146719–146743, 2020.

[49] E. Raffin, C. Wolinski, F. Charot, K. Kuchcinski, S. Guyetant, S. Chevobbe, and E. Casseau.
Scheduling, binding and routing system for a run-time reconfigurable operator based multimedia architecture.
In *DASIP*, pages 168–175, 2010.

References VI

[50] SambaNova.
Accelerated Computing with a Reconfigurable Dataflow Architecture.
SambaNova systems, 06 2021.

[51] Frank Schirrmeister.
Towards Decarbonization: Keeping Electronics Energy Consumption In Check, November 2021.

[52] Simon Schulz, Oliver Bringmann, Thomas Schweizer, and Wolfgang Rosenstiel.
Rotated parallel mapping: A novel approach for mapping data parallel applications on cgras.
In *2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14)*, pages 1–6, 2014.

[53] H. Singh, Ming-Hau Lee, Guangming Lu, F.J. Kurdahi, N. Bagherzadeh, and E.M. Chaves Filho.
Morphosys: an integrated reconfigurable system for data-parallel and computation-intensive applications.
IEEE Transactions on Computers, 49(5):465–481, 2000.

[54] Chilankamol Sunny, Satyajit Das, Kevin J. M. Martin, and Philippe Coussy.
Hardware based loop optimization for cgra architectures.
In Steven Derrien, Frank Hannig, Pedro C. Diniz, and Daniel Chillet, editors, *Applied Reconfigurable Computing. Architectures, Tools, and Applications*, pages 65–80, Cham, 2021. Springer International Publishing.

[55] Cheng Tan, Chenhao Xie, Ang Li, Kevin J. Barker, and Antonino Tumeo.
Aurora: Automated refinement of coarse-grained reconfigurable accelerators.
In *2021 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1388–1393, 2021.

[56] G. Theodoridis, D. Soudris, and S. Vassiliadis.
A Survey of Coarse-Grain Reconfigurable Architectures and Cad Tools, pages 89–149.
Springer Netherlands, Dordrecht, 2007.

[57] Kanishkan Vadivel, Mark Wijtvliet, Roel Jordans, and Henk Corporaal.
Loop overhead reduction techniques for coarse grained reconfigurable architectures.
In *2017 Euromicro Conference on Digital System Design (DSD)*, pages 14–21, 2017.

[58] Jian Weng, Sihao Liu, Vidushi Dadu, Zhengrong Wang, Preyas Shah, and Tony Nowatzki.
Dsagen: Synthesizing programmable spatial accelerators.
In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 268–281, 2020.

[59] Dhananiaya Wijerathne, Zhaoying Li, Anuj Pathania, Tulika Mitra, and Lothar Thiele.
Himap: Fast and scalable high-quality mapping on cgra via hierarchical abstraction.
In *2021 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1192–1197, 2021.

References VII

[60] M. Wijtvliet, L. Waeijen, and H. Corporaal.
Coarse grained reconfigurable architectures in the past 25 years: Overview and classification.
In *2016 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS)*, pages 235–244, 2016.

[61] Shouyi Yin, Dajiang Liu, Leibo Liu, Shaojun Wei, and Yike Guo.
Joint affine transformation and loop pipelining for mapping nested loop on CGRAs.
In *DATE*, pages 115–120, March 2015.

[62] Jonghee W. Yoon, Aviral Shrivastava, Sanghyun Park, Minwook Ahn, and Yunheung Paek.
A graph drawing based spatial mapping algorithm for coarse-grained reconfigurable architectures.
IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 17(11):1565–1578, 2009.

[63] Baofen Yuan, Jianfeng Zhu, Xingchen Man, Zijiao Ma, Shouyi Yin, Shaojun Wei, and Leibo Liu.
Dynamic-ii pipeline: Compiling loops with irregular branches on static-scheduling cgra.
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, pages 1–1, 2021.

[64] Yaqi Zhang, Nathan Zhang, Tian Zhao, Matt Vilim, Muhammad Shahbaz, and Kunle Olukotun.
Sara: Scaling a reconfigurable dataflow accelerator.
In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, pages 1041–1054, 2021.

[65] Zhongyuan Zhao, Weiguang Sheng, Qin Wang, Wenzhi Yin, Pengfei Ye, Jinchao Li, and Zhigang Mao.
Towards higher performance and robust compilation for cgra modulo scheduling.
IEEE Transactions on Parallel and Distributed Systems, 31(9):2201–2219, 2020.