Evaluation of SU(3) smearing on FPGA accelerator cards

Salvatore Cali[*,+], Grzegorz Korcyl[†] and Piotr Korcyl[*]

[†] Institute of Applied Computer Science, Jagiellonian University, ul. Łojasiewicza 11, 30-348 Kraków, Poland
[*] Institute of Theoretical Physics, Jagiellonian University, ul. Łojasiewicza 11, 30-348 Kraków, Poland
[+] Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

calis@mit.edu, grzegorz.korcyl@uj.edu.pl, piotr.korcyl@uj.edu.pl

UNIWERSYTET JAGIELLOŃSKI W KRAKOWIE

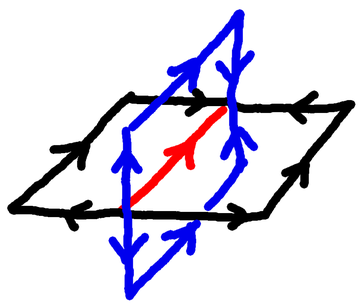
Massachusetts Institute of Technology

1. Introduction

Recent FPGA accelerator cards promise large acceleration factors for some specific computational tasks. In the context of Lattice QCD calculations, we investigate the possible gain of moving the $SU(3)$ gauge field smearing routine to such accelerators. We study Xilinx Alveo U280 cards and use the associated Vitis high-level synthesis framework. We discuss the possible pros and cons of such a solution based on the gathered benchmarks.

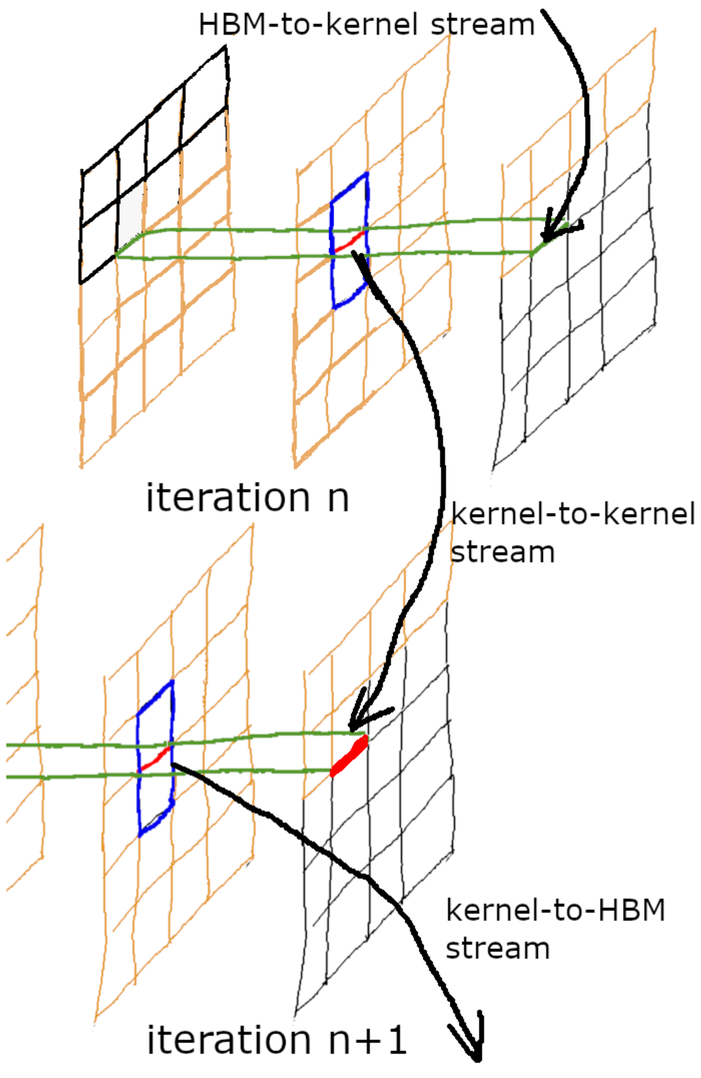
2. APE smearing stencil

APE smearing stencil averages neighbouring, parallel gauge links in an gauge invariant manner by evaluating 6 staples matrices and averaging them with some appropriate weights, as sketched on the figure:



We assume that the gauge links have been transferred from the host to the HBM memory. Next, the links are streamed in the same order to the programmable logic, where they are transformed and stored back in the HBM memory. This process can be iterated. Ultimately, the smeared gauge field is transferred back to the host.

3. Pipelined and streamlined design



Schematic view of data flow in the HBM-kernel-kernel-HBM stream with cyclic buffers (orange) implemented in the U/BRAM

4. Resource consumption

Composition of multiply_by_staple function

component	#	latency	interval	DSP	FF	LUT
grp_compute_staple_forward_fu	3	39	2	400	31231	21159
grp_compute_staple_backward_fu	3	39	2	400	31231	21159
grp_add_two_fu	3	4	1	36	2827	2178

Comparison of resource consumption of various compute kernels for different data types compiled for U280 card at 300 MHz with Vitis HLS 2020.2 (resources in % of total / of one SLR)

function	prec.	latency	II	BRAM	DSP	FF	LUT
compute_staple_forward	double	65	2	0	12 / 37	5 / 16	6 / 18
compute_staple_forward	double	67	4	0	6 / 18	3 / 10	3 / 9
compute_staple_forward	double	71	8	0	3 / 9	2 / 6	1 / 5
multiply_by_staple	double	90	2	0	77 / 231	34 / 103	39 / 118
multiply_by_staple	double	93	4	0	38 / 115	21 / 63	21 / 63
multiply_by_staple	double	99	8	0	19 / 57	13 / 41	11 / 34
compute_staple_forward	float	69	2	0	5 / 16	2 / 7	2 / 7
compute_staple_forward	float	72	4	0	2 / 8	1 / 4	1 / 4
multiply_by_staple	float	100	2	0	17 / 104	15 / 47	16 / 50
multiply_by_staple	float	105	4	0	17 / 52	10 / 30	9 / 29
compute_staple_forward	half	72	2	0	4 / 13	1 / 4	1 / 4
multiply_by_staple	half	103	2	0	27 / 82	10 / 31	9 / 29
su3_projection	double	869	8	0	14 / 43	10 / 31	8 / 26
su3_projection	float	899	4	0	13 / 39	7 / 23	7 / 23
su3_projection	half	909	2	0	20 / 62	8 / 24	7 / 23
full	double	989	8	0	33 / 100	25 / 75	20 / 62
full	float	1022	4	0	30 / 91	17 / 53	17 / 53
full	half	1037	4	0	24 / 73	11 / 35	11 / 35
full	half	1014	2	0	49 / 147	19 / 57	17 / 53

5. Timings and performance

Smearing of one gauge link (9 complex floating point numbers) requires:

- $18 \times 9 = 162$ floating point memory loads,
- $324 \times 12 + 108 = 3996$ FLOPs.

and 2790 FLOPs for the $SU(3)$ projection which we do with 4 iterations following JHEP0508 (2005) 051.

The HBM memory on the Xilinx U280 card has 32 512-bit wide ports which can run at 300 MHz. One data package contains $4 \times 18 = 72$ floating point numbers.

	total size in bits	# 512-bit words	initiation interval
double	4608	9	2 - 9
float	2304	4.5	2 - 5
half	1652	2.25	2 - 3

The initiation interval inferred from the HBM-programmable logic memory bandwidth sets the performance limit on a single kernel.

precision	II	staples [GFLOP/s]	[GFLOP/s]
double	8	150	105
float	4	300	210
half	2	600	420

Estimated total performance assuming one kernel (smearing of gauge links in one direction) in each of the 3 SLR

precision	# kernels	[GFLOP/s]
double	3	765
float	3	1530
half	3	3060

6. Conclusions

- efficient single kernels but difficult to route and place including cyclic buffers
- need to optimize data flow infrastructure by hand to improve routing and minimize congestion problems at the memory banks

7. Acknowledgements

Work supported Foundation for Polish Science grant no. TEAM/2017-4/39, by the Polish Ministry for Science and Higher Education grant no. 7150/E-338/M/2018, and by the Priority Research Area Digiworld under the program Excellence Initiative – Research University at the Jagiellonian University We gratefully acknowledge hardware donations from Xilinx within the Xilinx University Program.