GPU and network innovations

John W. Romein

Motivation

• goals:

- more powerful instruments ← higher bandwidth
- improve energy efficiency
- reduce implementation effort





Agenda

•

integrated approach

GPU innovation •

Tensor cores

- hardware matrix-multiplication units
 - limited-precision input data
 - 5-10x faster than regular GPU cores
- accelerates deep learning
- since ~5 years

	_	L0 In	in line													
		and the second division of the second divisio	ISKIUC	tion C	ache			L0 Instruction Cache								
	War	p Sch	edule	r (32 ti	hread	V AT			Warp Scheduler (32 thread/clk)							
	Di	spatch	h Unit	(32 th	rear o	:lk)				Dis	spatch	1 Unit	(32 th	read/o	:lk)	
	Reg	ister	File (1	16,384	x 32	?-bit)				Reg	ister	File (1	16,38	4 x 32	-bit)	
FP64	INT	INT	FP32	FP3	T			F	P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FP 2				F	P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FF 12				F	P64	INT	INT	FP32	FP32			TENSOR CORE
FP64	INT	INT	FP32	FI 32	TEN	SOR	TENSOR	F	P64	INT	INT	FP32	FP32	TEN	SOR	
FP64	INT	INT	FP32	FI 12	CC	ORE	CORE	F	P64	INT	INT	FP32	FP32	CO	RE	
FP64	INT	INT	FP32	FF 12				F	P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FP 2				F	P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FP3				F	P64	INT	INT	FP32	FP32			
LD/ LD/ ST ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	SFU	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	LD/ ST	SFU
	Di	spatch	n Unit	(32 th	read/o	lik)				Dis	spatch	1 Unit	(32 th	read/o	:lk)	
	Reg	Ister	rile (1	10,304	+ x 32	(-DIL)				Reg	ister	rile (10,30	+ x 32	-DIL)	
FP64	INT	INT	FP32	FP32				F	P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FP32				F	P64	INT	INT	FP32	FP32			TENSOR CORE
FP64	INT	INT	FP32	FP32				F	P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FP32	TEN	ISOR	TENSOR	F	P64	INT	INT	FP32	FP32	TEN	SOR	
FP64	INT	INT	FP32	FP32				F	P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FP32					P64	INT	INT	FP32	FP32			
FP64	INT	INT	FP32	FP32					P64	INT	INT	FP32	FP32			
LD/ LD/	LD/	LD/	LD/	LD/	LD/	LD/	SEII	LDV	LD/	LD/	LD/	LD/	LD/	LDV	LD/	9511
	ST	ST	ST	ST	ST	ST	STU	ST	ST	ST	ST	ST	ST	ST	ST	SFU
ST ST			-		_	_		and have a							_	



Use tensor cores for signal processing?

- yes, if:
 - algorithm translates to matrix-matrix multiplications
 - correlator: <
 - beam former: 🗸 , 🗙
 - FIR filter: (likely) X
 - non-uniform Fourier transform:
 - FFT: 🗙 (radix 8: 🗸)
 - operates on few bits \checkmark



The Tensor-Core Correlator¹

• GPU library

BLOCKS

- performs (tensor-core) computations
- not full application (no I/O etc.)
- highly optimized
- hides nasty details
- open source²
- rapidly adopted by radio telescopes worldwide

1) J.W. Romein, The Tensor-Core Correlator, A&A 656(A52), Dec 2021

2) https://git.astron.nl/RD/tensor-core-correlator



Implementation challenges

BLOCKS

not supported by tensor cores

- complex numbers
 triangular output format _
- 3) fast data fetching
- 4) 4-bit mode requires ptx assembly hacking
- all hidden from the user





June 5th, 2024	, 2024	5tł	lune	
----------------	--------	-----	------	--





NVIDIA A100 PCIe 40 GB

June 5th, 2024



Tensor cores: answer to the demise of Moore's law



• historical best-case correlator performance



The I/O challenge

- GPU correlators in last decade: up to 100x performance
- I/O must scale proportionally
- receiving >40 Gb/s data: prohibitive OS overhead
 - wait for faster processors
 - need new I/O techniques



Data Transport



- digitizer \rightarrow corner turn \rightarrow correlator
- stream data: FPGA \rightarrow switch \rightarrow GPU



Data Transport

explore new methods ۲

BLOCKS

- Data Plane Development Kit
- implementing demo correlator •



Remote Direct Memory Access (RoCE v2)
 Data Plana Dovelopment Kit
 different advantages



Data Plane Development Kit



- Θ application controls NIC \rightarrow no OS overhead
- application sends/receives Ethernet packets
- new: GPU support \rightarrow GPU handles packets



DPDK demo correlator

- correlator *partially* implemented
 - -packet receipt 🗸

- -delay input streams X
- -GPU filtering 🗙
- −GPU correlator ✓
- writing correlations \checkmark





- most powerful GPU ever
- 14x more CPU ↔ GPU bandwidth than PCIe gen4



Grace Hopper systems setup



- all PCIe slots filled with NICs
 - 1200 Gb/s total



Grace Hopper DPDK results



- result: 1196 Gb/s received in GPU memory
 - DPDK (software innovation): $40 \rightarrow 199$ Gb/s (PCIe gen4 NIC/GPU)
 - Grace Hopper (hardware innovation): 199 \rightarrow 1196 Gb/s



To do

- finish the demo application
 - buffer input data to apply delays
 - properly channelize data



Motivation revisited

• goals:

BLOCKS

- more powerful instruments ← higher bandwidth *promising results*
- improve energy efficiency tensor cores:

DPDK: okay, but RDMA may be better

- reduce implementation effort *complex, but hidden inside Radio Block*

Summary

• ~10 years:

- innovations like tensor cores \rightarrow up to 100x performance
- compute bound \rightarrow I/O bound
- integrated GPU / network approach looks very promising