MoonGen

A Scriptable High-Speed Packet Generator

Paul Emmerich

March 4th, 2015

Chair for Network Architectures and Services
Department of Informatics
Technische Universität München

Agenda

State of the Art in Packet Generation

High-Speed Multi-Core Packet Processing with DPDK and LuaJIT

Hardware Timestamping on Commodity NICs

Precise Rate Control

State of the Art

- Hardware Packet Generators
  - (+) Precise & accurate (timestamps, rate control)
  - (+) Fast
  - (–) Inflexible
  - (–) Expensive

State of the Art

- Hardware Packet Generators
  - (+) Precise & accurate (timestamps, rate control)
  - (+) Fast
  - (–) Inflexible
  - (–) Expensive
- Software Packet Generators
  - (+) Run on cheap commodity hardware
  - (+) Some are fast
  - ( ) Some are flexible to a certain degree (e.g. Pktgen-DPDK, Ostinato)
  - (–) Imprecise (timestamps, rate control)
  - (–) Inaccurate (timestamps)

Design Goals

Design Goal of MoonGen

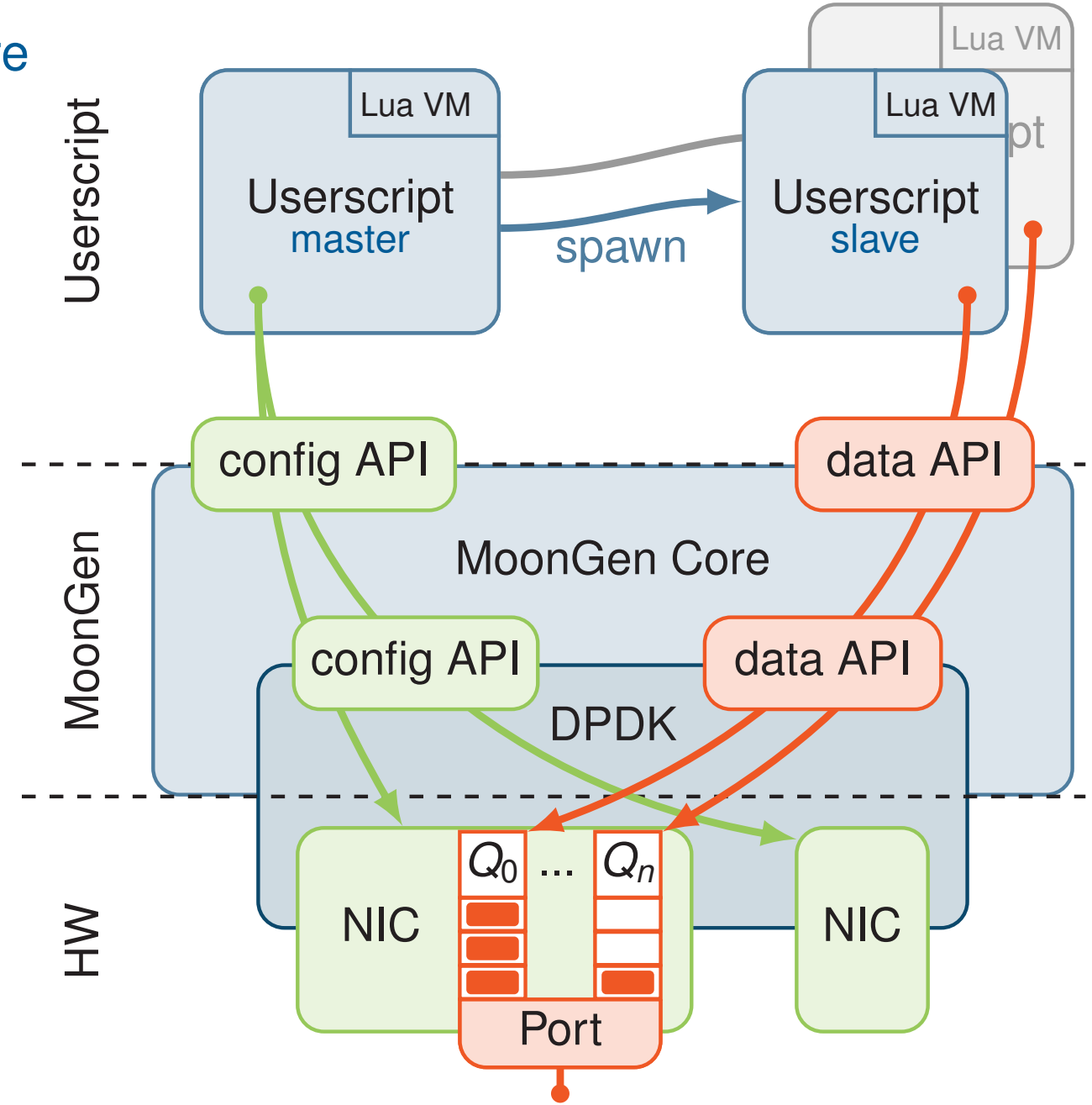
Combine the advantages of both approaches while avoiding their disadvantages.

Design Goals

Design Goal of MoonGen
Combine the advantages of both approaches while avoiding their disadvantages.

- Fast: DPDK for packet I/O, explicit multi-core support
- Flexible: Craft all packets in user-controller Lua scripts
- Timestamping: Utilize hardware features found on modern commodity NICs
- Rate control: Hardware features and a novel software approach

Architecture



Hardware Timestamping

- NICs support PTP for precise clock synchronization
- PTP support requires hardware timestamping capabilities
- These can be (mis-)used for delay measurements
- Typical precision
  - $\pm$6.4 ns (Intel 10 GbE chips)
  - $\pm$32 ns (Intel GbE chips)
- Some restrictions
  - Packets must be UDP or PTP L2 protocol
  - Minimum UDP packet size is 84 bytes

Hardware Timestamping Precision and Accuracy

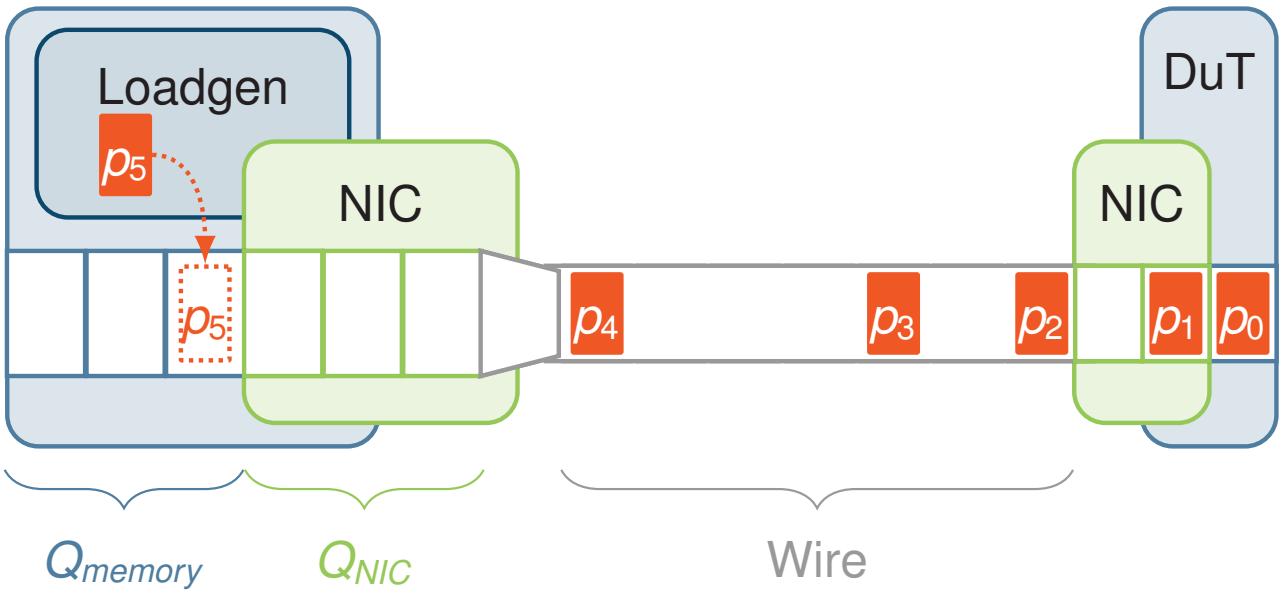
- Measure latencies of cables of various length
- Calculate coding time k and propagation speed v_p

NIC	t_{2m} [ns]	$t_{8.5m}$ [ns]	t_{10m} [ns]	t_{20m} [ns]	k [ns]	v_p
82599 (fiber)	320	352	-	403.2	310.7 ± 3.9	$0.72c \pm 0.056c$
X540 (copper)	-	-	2252.8	2310.4	2195.2 ± 9.6	$0.59c \pm 0.065c$

Timestamping Precision and Accuracy

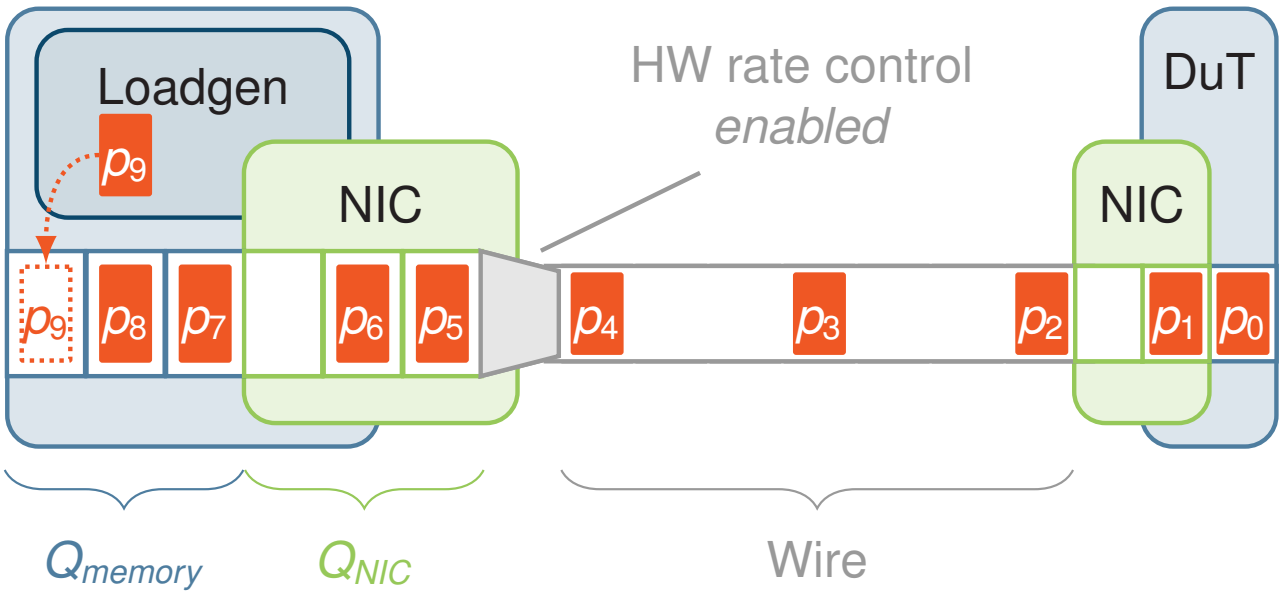
The linear behaviour and reasonable results for coding time and propagation speed show that MoonGen can measure latency with sub-microsecond precision and accuracy.

Software Rate Control in Existing Packet Generators



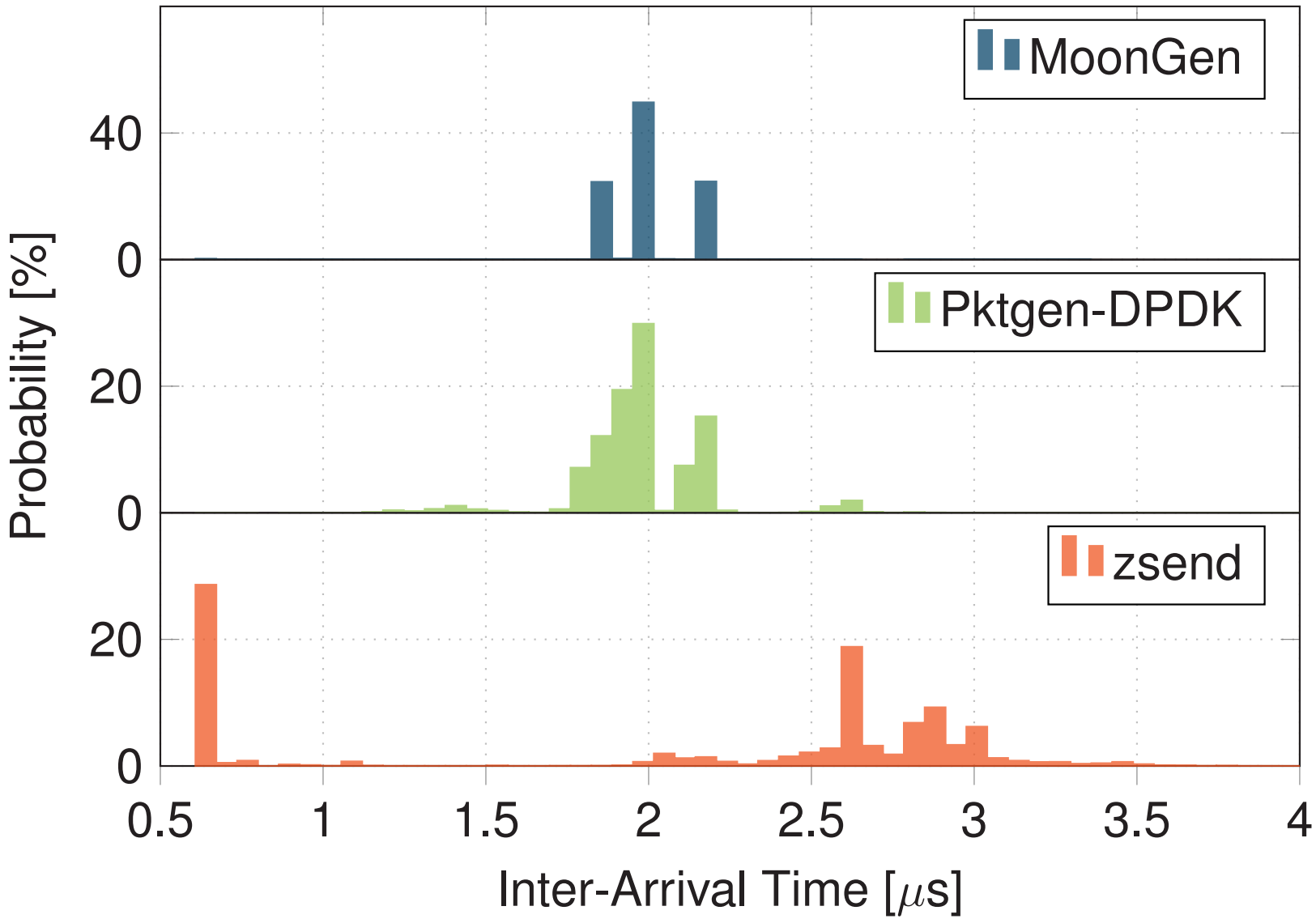
- Try to push single packets to the NIC
- Queues cannot be used, no batch processing
- NICs work with an asynchronous push-pull model
- Can lead to micro-bursts
- Unreliable, imprecise, and bad performance

Hardware Rate Control

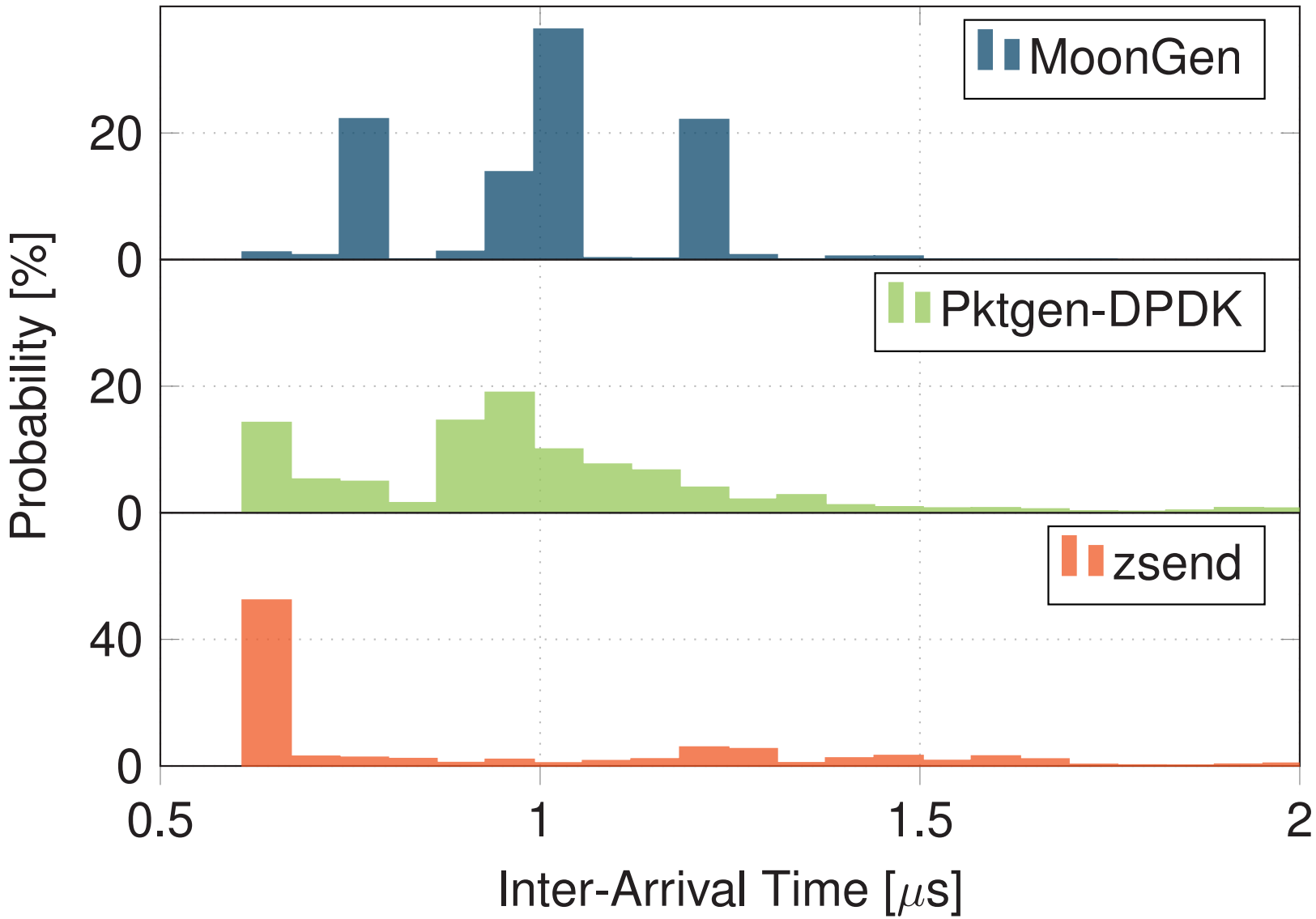


- Modern NICs support rate control in hardware
- Limited to constant bit rate and bursty traffic
- Precision controlled by the hardware
- High performance as queues can be used

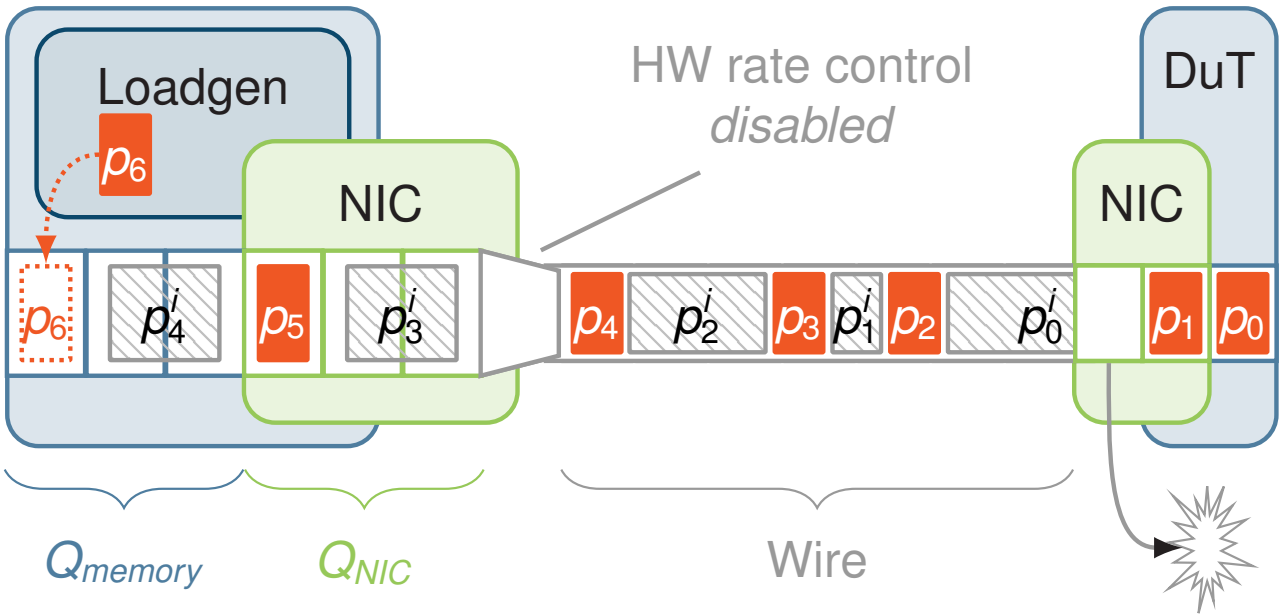
Evaluation: 500 kpps



Evaluation: 1,000 kpps



Software Rate Control Based on Invalid Packets



- Fill gaps with invalid packets p^i (e.g. bad CRC)
- NIC in the DuT drops invalid packets without side-effects
- Combines advantages of both approaches
- Precision limited by byte rate (0.8 ns) and minimum packet size (50 Byte)

Q & A

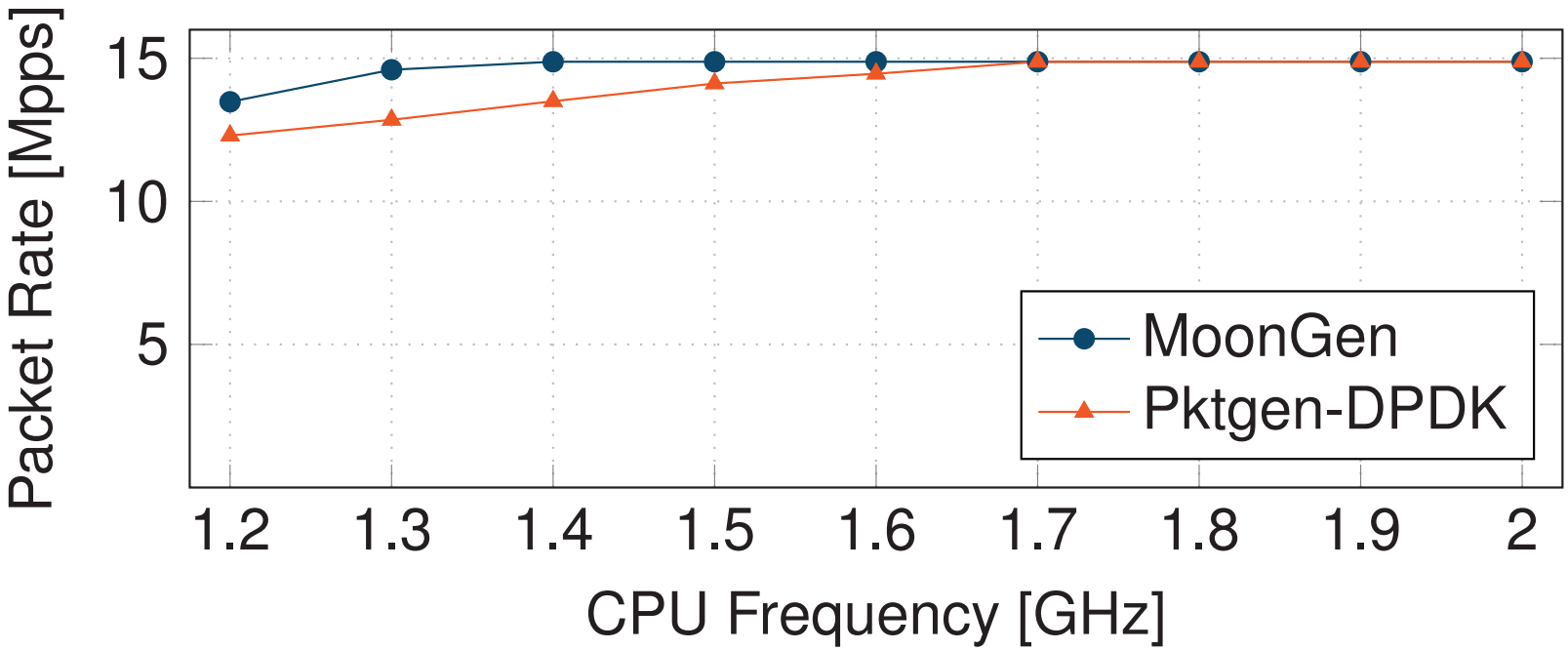
Try MoonGen yourself!

`https://github.com/emmericp/MoonGen`

Questions?

[Backup Slide] Performance I: Lua Can be Faster Than C

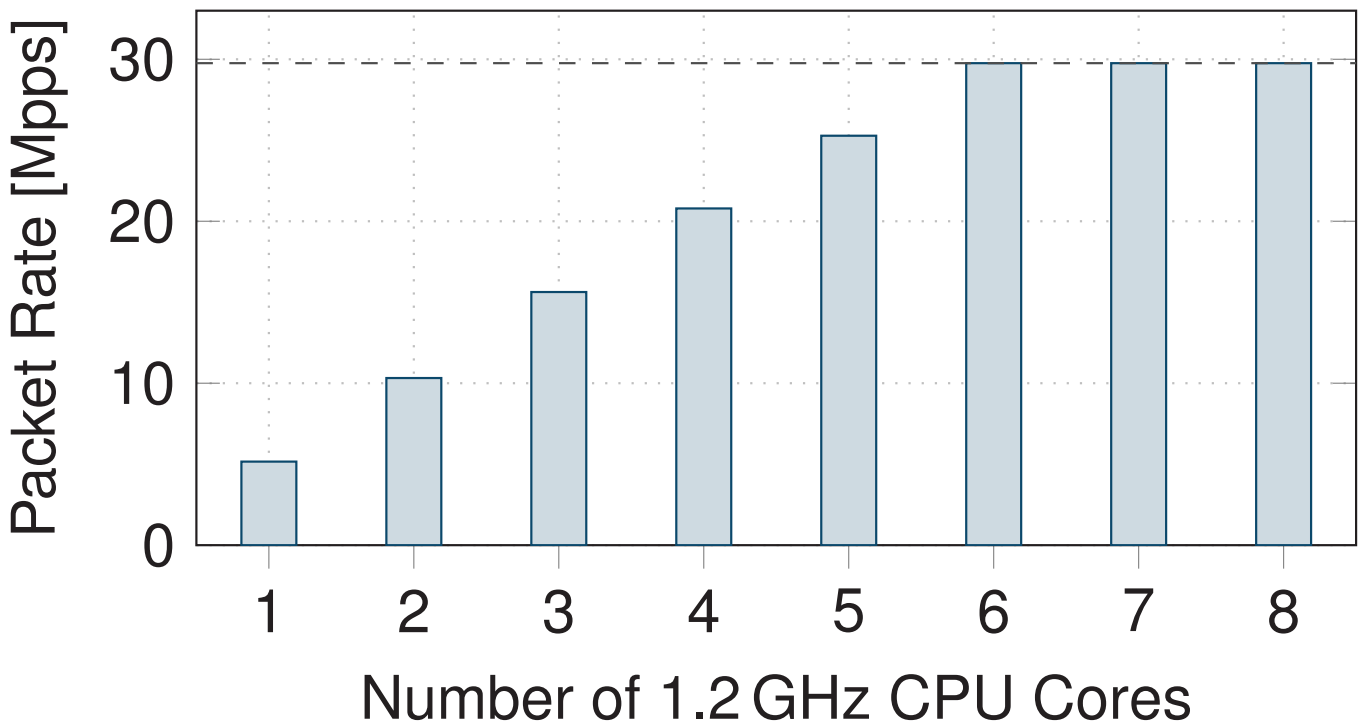
- UDP packets from varying source IP addresses



- Pktgen-DPDK needs a complicated main loop that covers all possibilites
- MoonGen can use a tight inner loop

[Backup Slide] Performance II: Heavy Workload and Multi-Core Scaling

- Generate random UDP packets on 2 10 GBit NICs
- 8 calls to Lua's standard `math.random` per packet
- CPUs artificially clocked down to 1.2 GHz

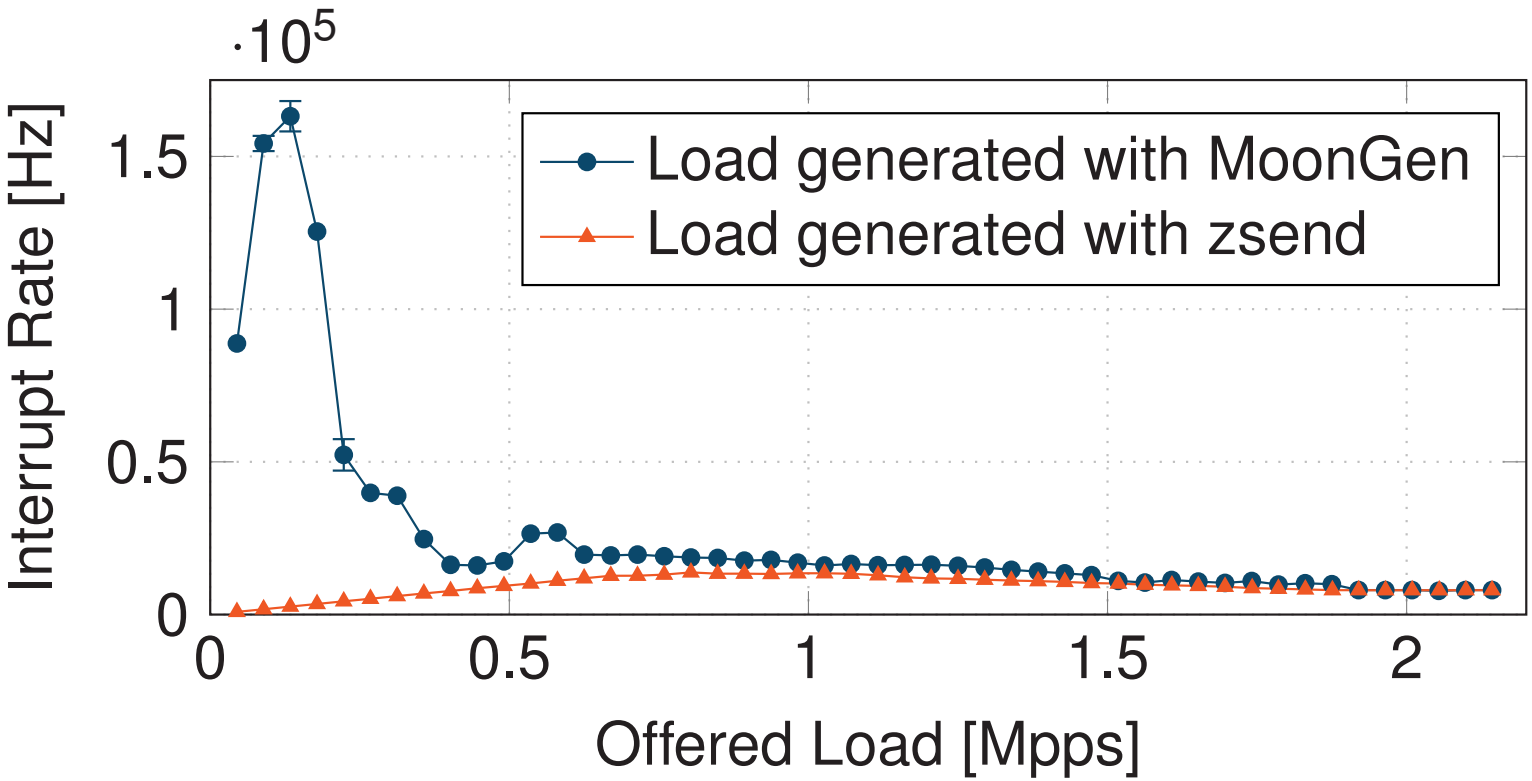


[Backup Slide] HW/SW Rate Control Details

Rate	Software	Bursts	±64 ns	±128 ns	±256 ns	±512 ns
500 kpps	MoonGen	0.02%	**49.9%**	**74.9%**	**99.8%**	**99.8%**
	Pktgen-DPDK	**0.01%**	37.7%	72.3%	92%	94.5%
	zsend	28.6%	3.9%	5.4%	6.4%	13.8%
1000 kpps	MoonGen	**1.2%**	**50.5%**	52%	**97%**	**100%**
	Pktgen-DPDK	14.2%	36.7%	**58%**	70.6%	95.9%
	zsend	52%	4.6%	7.9%	24.2%	88.1%

[Backup Slide] Effects of Bad Rate Control

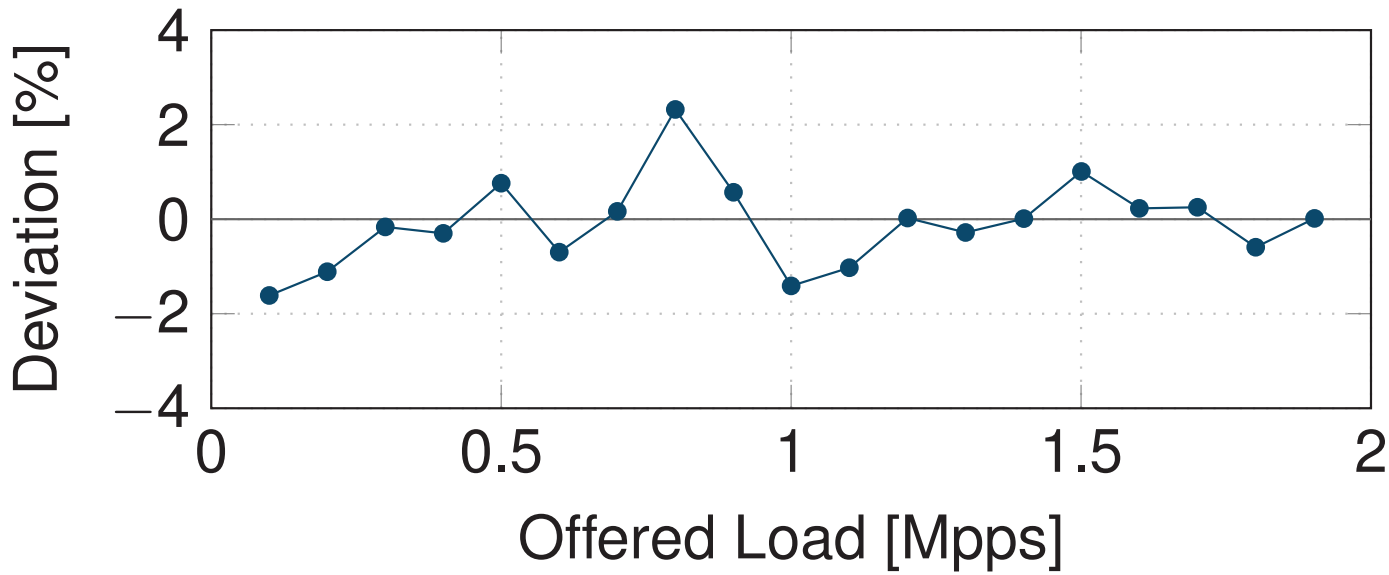
- Interrupt rate of an Open vSwitch packet forwarder



- Micro-bursts confuse dynamic interrupt throttling
- This affects latency (cannot be measured with zsend)

[Backup Slide] Effects of Invalid Packets

- Median latency of an Open vSwitch packet forwarder
- Packet rate controlled by hardware vs. invalid frames



- Minor modifications to the DuT (e.g. an active SSH session) result in a deviation of up to 15% with the same rate control mechanism

[Backup Slide] Poisson Traffic