Leveraging Mobile GPUs for Flexible High-speed Wireless Communication

Qi Zheng, Cao Gao, Trevor Mudge, Ronald Dreslinski

*University of Michigan, Ann Arbor

The 3rd International Workshop on Parallelism in Mobile Platforms (PRISM-3)

Jun 14, 2015

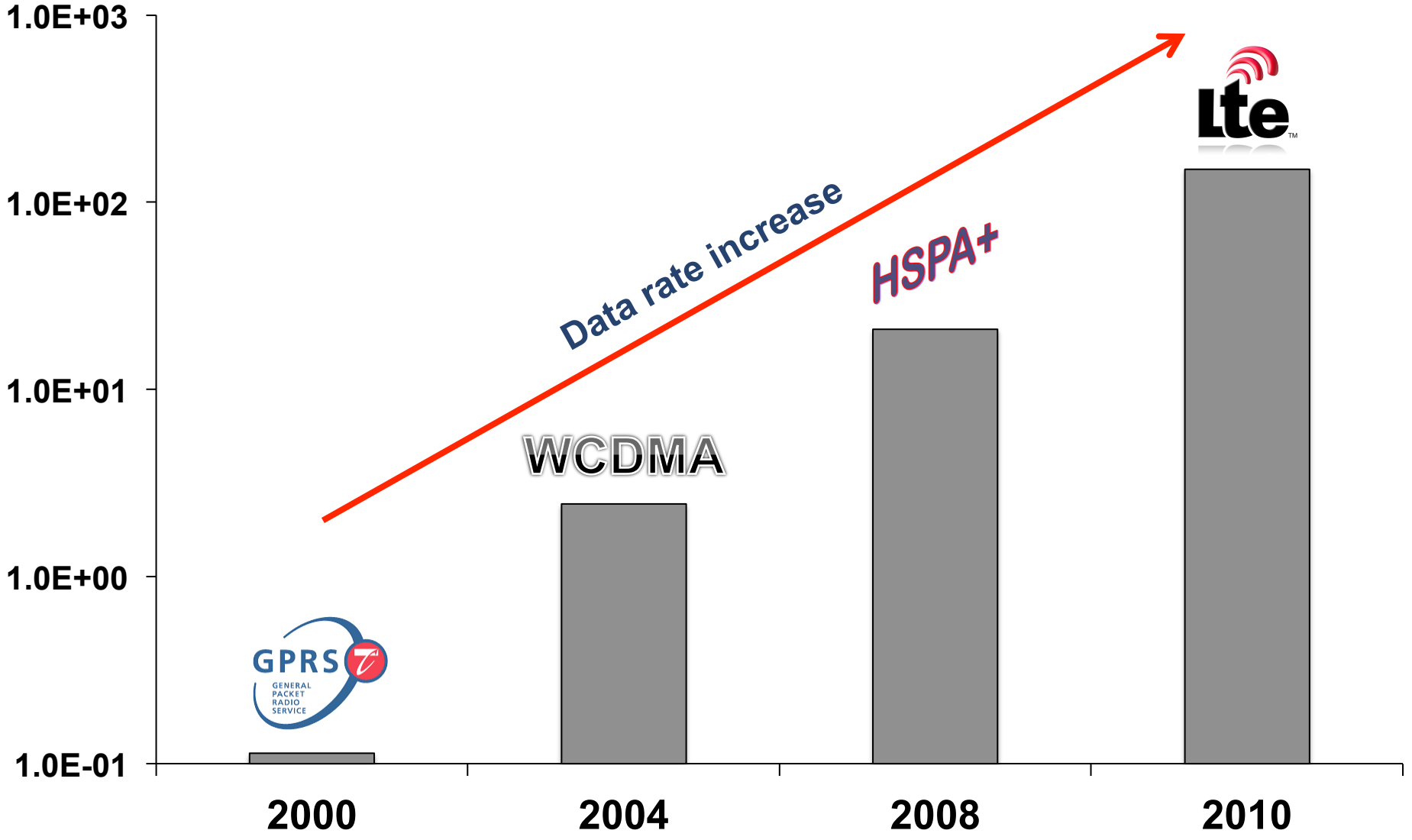
Popular Mobile Device & Applications

Google

facebook

amazon.com

Fast Technology Evolution



Support of Multiple Standards

GSM®
GLOBAL SYSTEM FOR MOBILE COMMUNICATIONS

GPRS
GENERAL PACKET RADIO SERVICE

CDMA2000
Leading in Mobile Technology and Service Innovation

WCDMA

HSPA+

wimax

Lte™

Wireless Processor in Mobile Devices

- Hardware accelerator based processors
  - ASIC
  - DSP with many ASICs
  - FPGA customized as ASICs

✓High performance

✓High energy efficiency

✕Long time to market

✕Hard and costly to maintain compatibility

Software-defined Radio!

Software-defined Radio (SDR)

- Wireless system is implemented entirely in software
  - General-purpose programming language

✓Very short time to market

- Only implement new software for new technologies

✓Easy to maintain compatibility

- Different programs for different standards

? Performance

? Energy efficiency

Processor for SDR

CDMA2000
Leading in Mobile Technology and Service Innovation

Lte™

TD-S

GPU!

- High computing throughput
- High energy efficiency
- Explore DLP and TLP
- Good programmability

Graphics Processing Unit (GPU)

- High throughput:
  - GOPS-level peak throughput
- SIMT execution model
  - Make use of abundant DLP and TLP
- Good programming support
  - CUDA/OpenCL

Desktop GPU
Power ≥ 100W

Mobile GPU
Power ~ 1W

Our Contribution

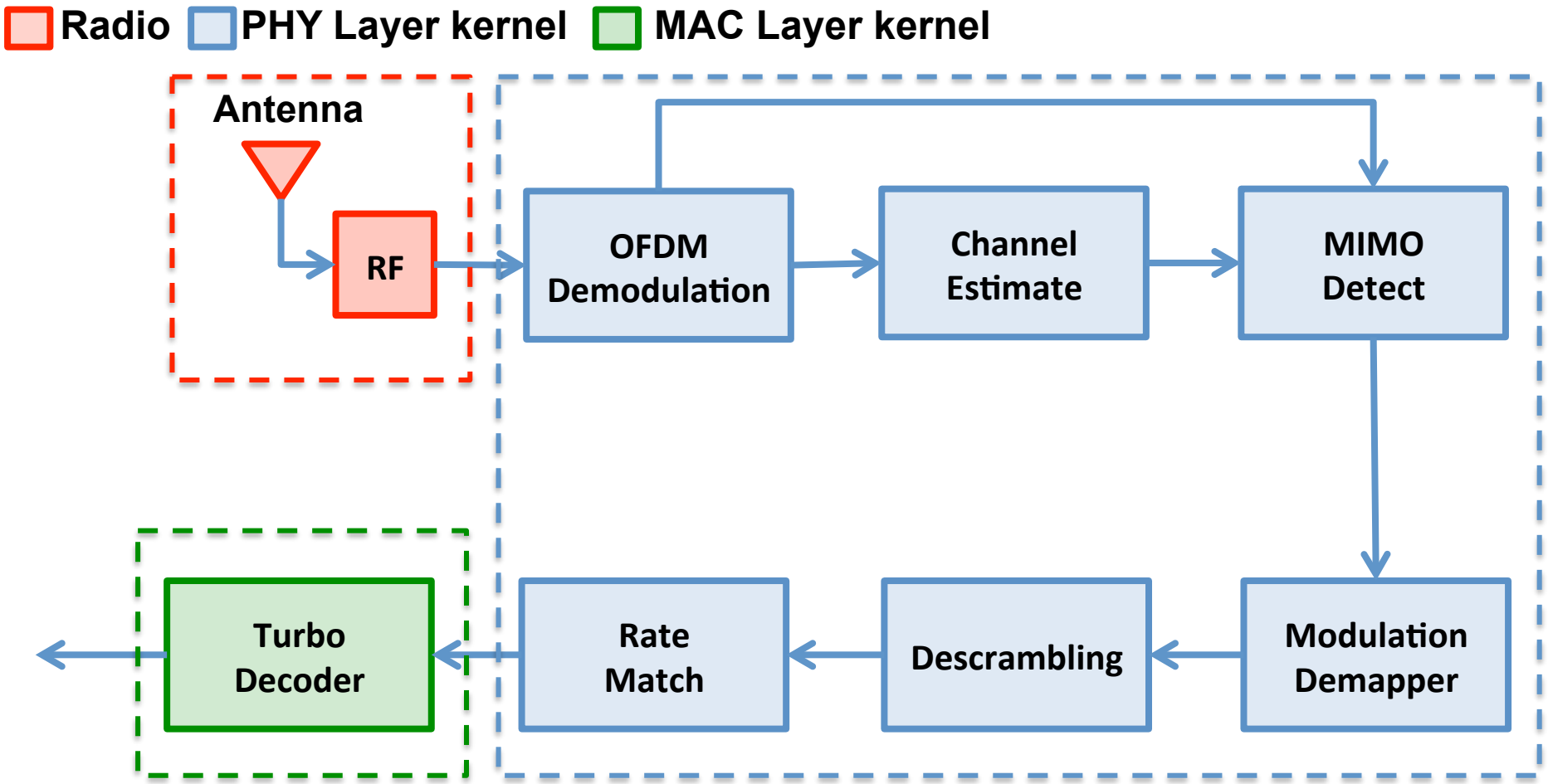
- Parallel implementations of LTE baseband kernels on a mobile GPGPU
- Performance of running LTE baseband
  - Meet the specification peak data rate for physical layer kernels
  - Get close to 10Mpbs typical data rate with for the Turbo decoder
  - Meet 20ms latency budget
- Power and energy efficiency of a mobile GPGPU
  - High power compared with application-specific processors
  - Still in an acceptable range
  - Implications of GPU architecture to further reduce the power

Outline

- Motivation
- **LTE baseband kernels**
- Running LTE baseband on a mobile GPGPU
- Implications for future mobile GPGPUs
- Conclusion

LTE Baseband Downlink Receiver

- Downlink Receiver has the most computations in a mobile device



Parallelism in Each Kernel

- The PHY layer kernels
  - Antenna-level
  - Symbol-level
  - Subcarrier-level
  - Algorithm-level
- The Turbo Decoder
  - Trellis-level
  - Subblock-level
- For all kernels
  - Batch multiple packets

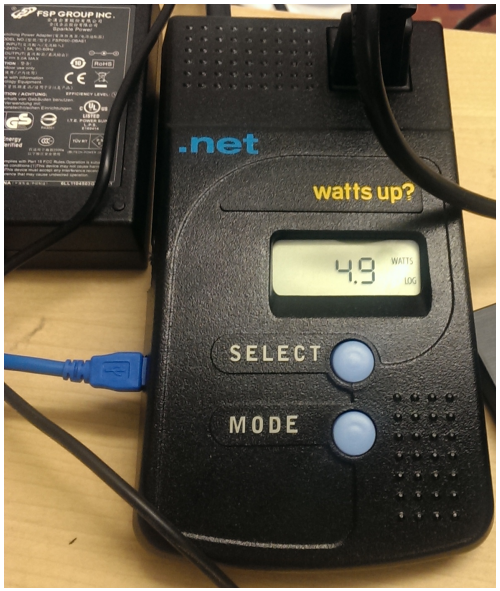
Outline

- Motivation
- LTE baseband kernels
- **Running LTE baseband on a mobile GPGPU**
  - **Methodology**
  - PHY layer
  - Turbo decoder
  - Power and energy efficiency
- Implications for future mobile GPGPUs
- Conclusion

Experimental Setup

- Jetson TK1 Board
  - NVIDIA 4-Plus-1™ quad-core ARM® Cortex-A15 CPU
  - **NVIDIA Kepler GK20a GPU**
    - **192 CUDA cores**
    - **≤ 852MHz**
    - **256 KB RF / 64 KB L1 memory / 128 KB L2 cache**
  - 2G DDR3L RAM (shared by CPU and GPU)

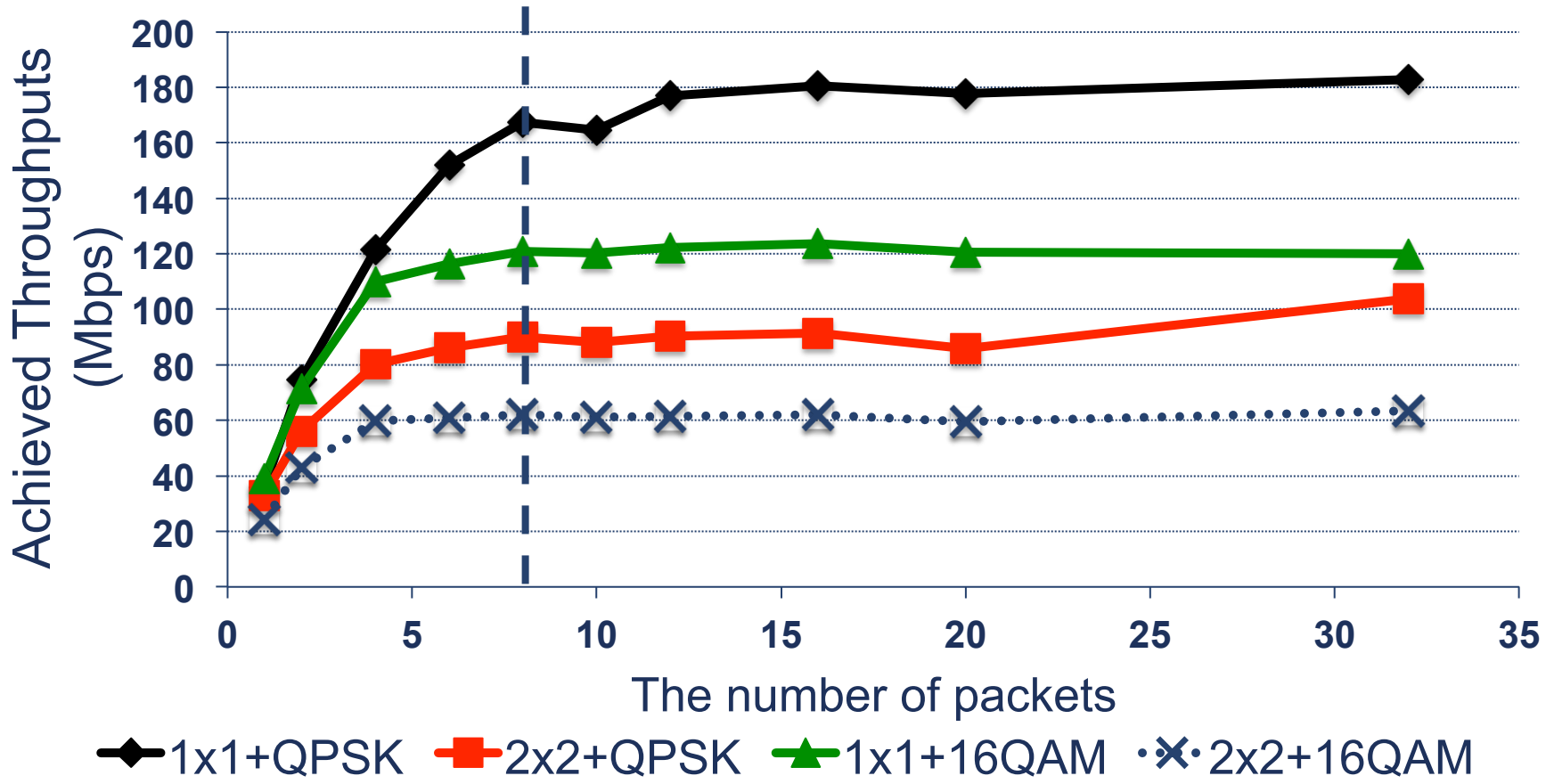
- Wattsup.net Power Meter
  - 33% board power consumed by SoC



Outline

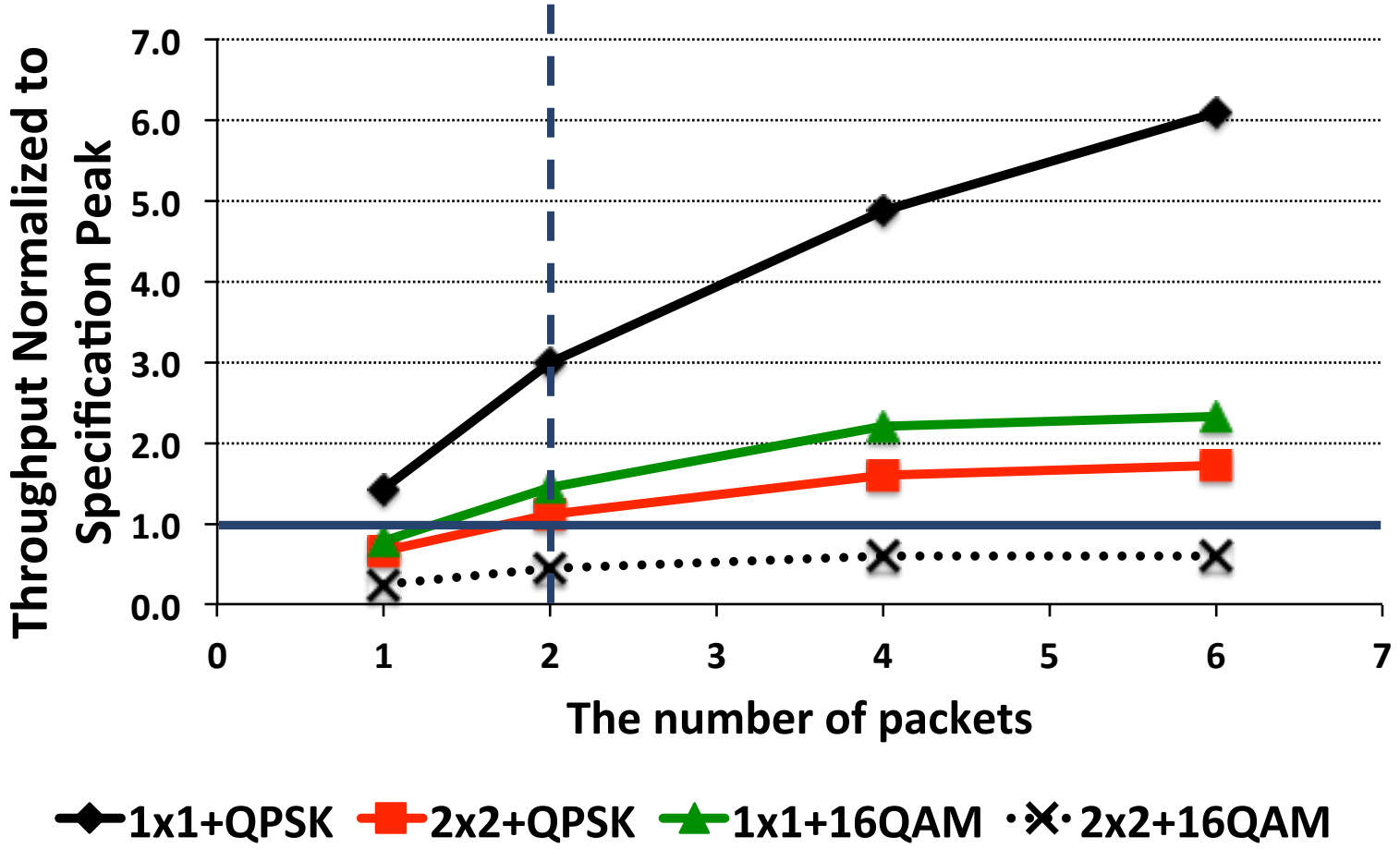
- Motivation
- LTE baseband kernels
- **Running LTE baseband on a mobile GPGPU**
  - Methodology
  - **PHY layer**
  - Turbo decoder
  - Power and energy efficiency
- Implications for future mobile GPGPUs
- Conclusion

PHY Layer Throughput



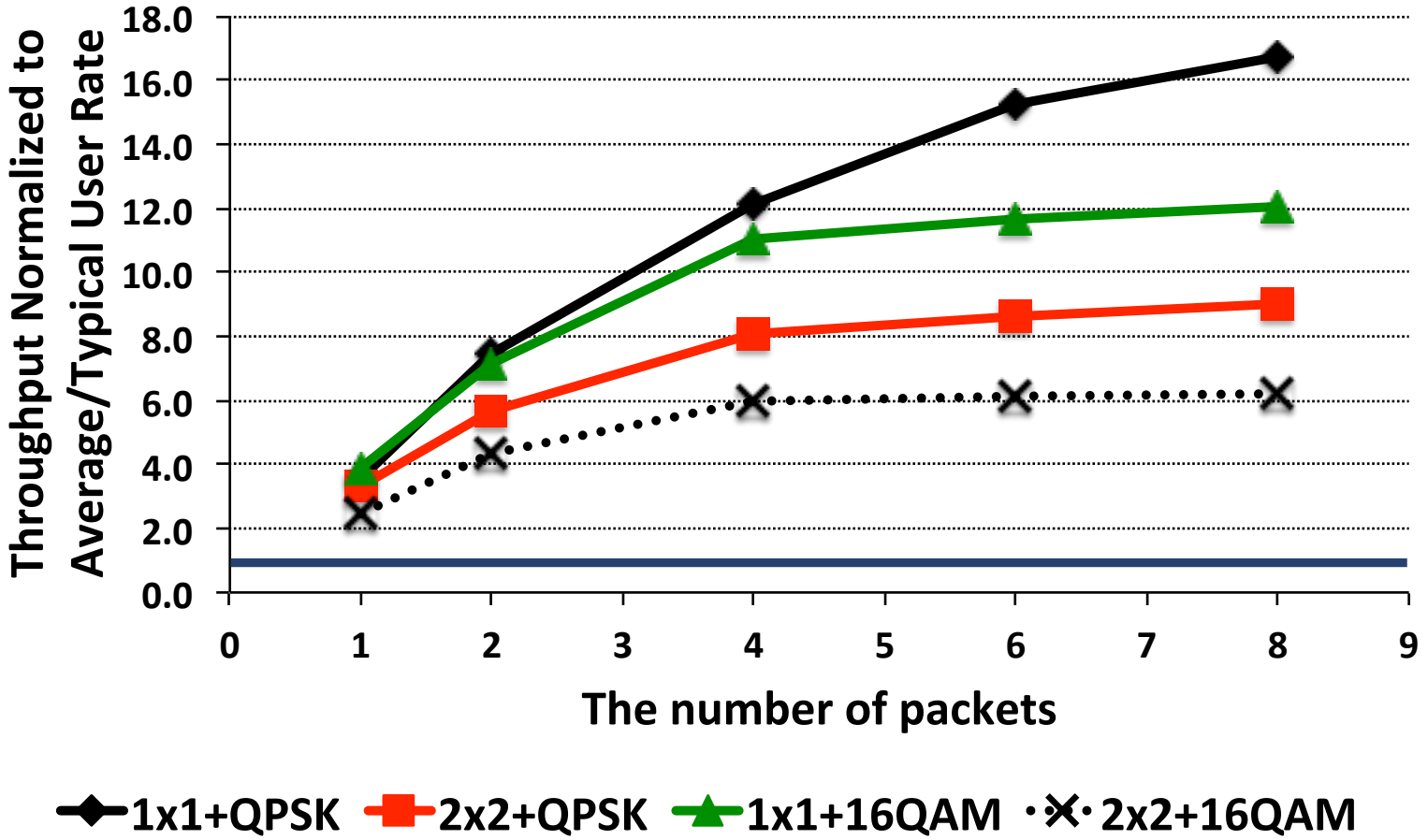
Packet batching increases throughput
Saturate at 8

Compare with Specification Peak



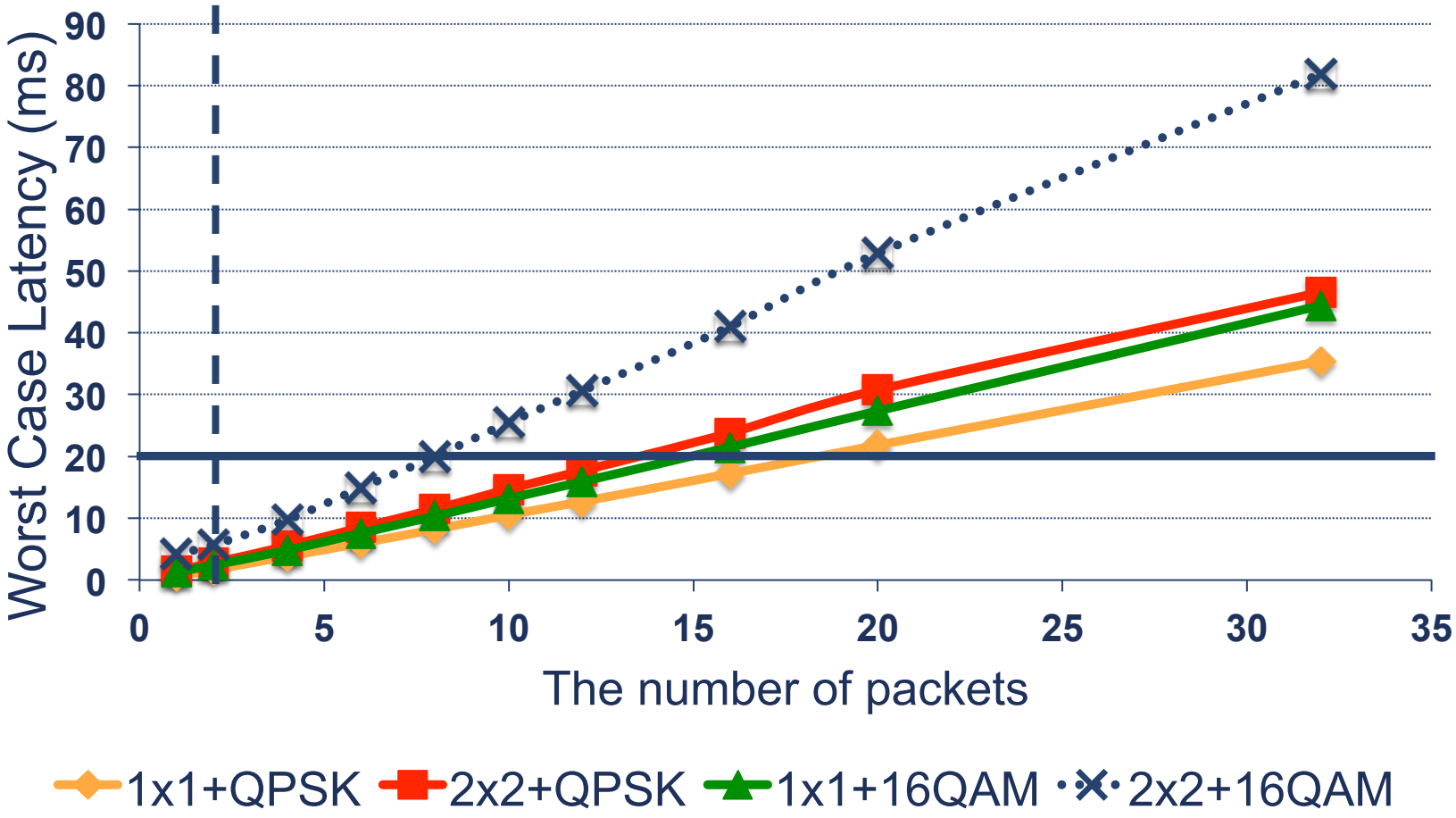
Overall, 2 packets -- a good batching size

Compare with 10Mbps Typical Rate



Fulfill all typical user data rates

PHY Layer Latency

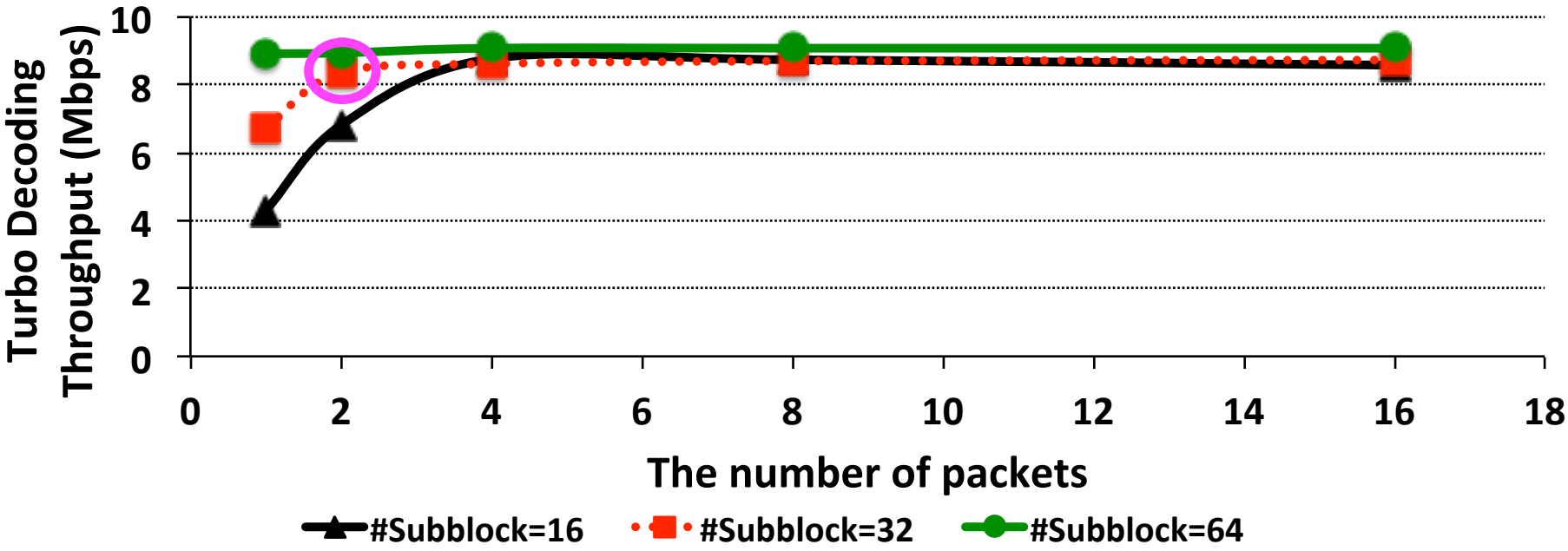


Packet batching also increases latency,
can batch ≤ 8 packets given 20ms budgets

Outline

- Motivation
- LTE baseband kernels
- **Running LTE baseband on a mobile GPGPU**
  - Methodology
  - PHY layer
  - **Turbo decoder**
  - Power and energy efficiency
- Implications for future mobile GPGPUs
- Conclusion

Turbo Throughput



Throughput increases with #subblocks an batching sizes

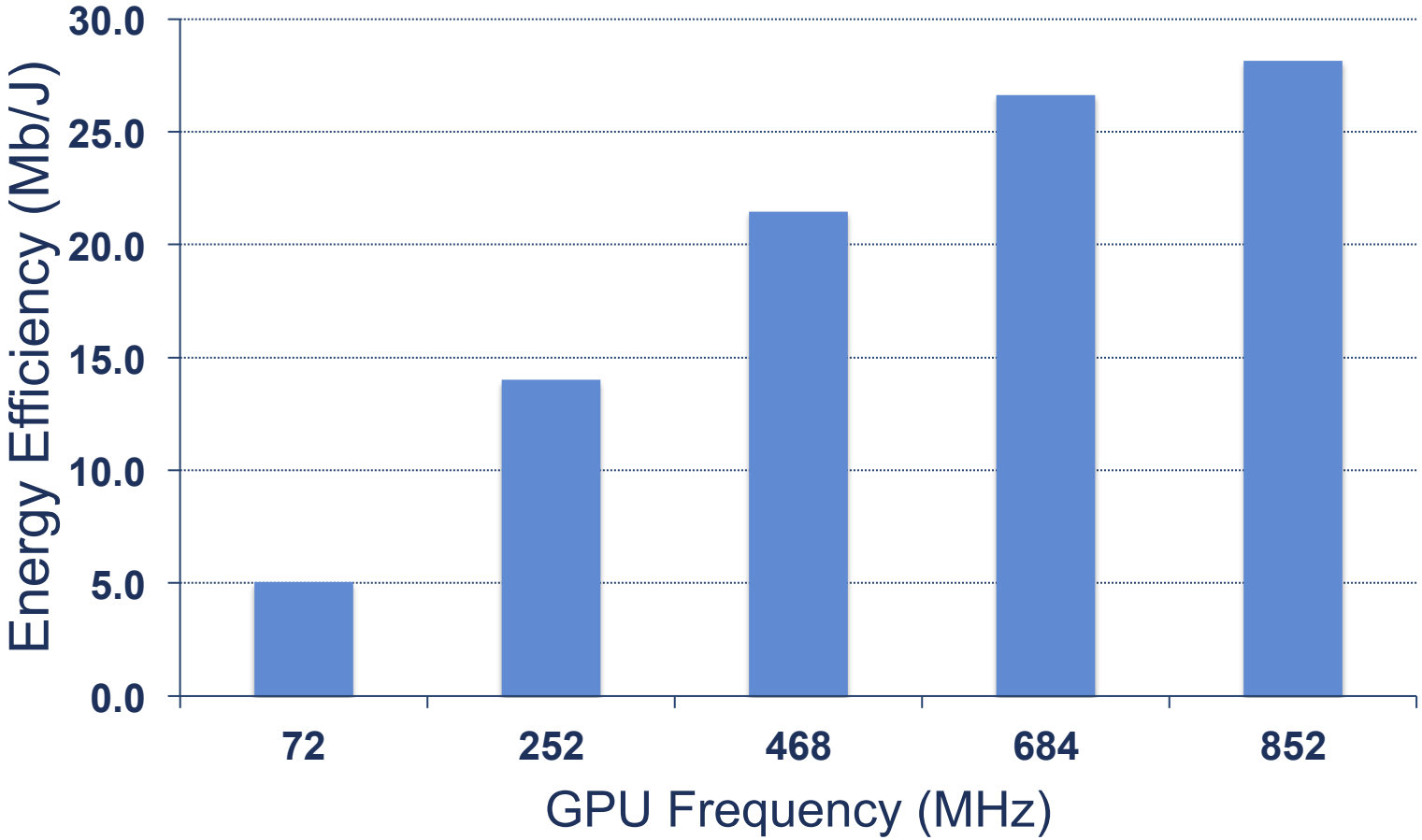
32 subblocks + **2** packets:
8.4 Mpbs throughput, ***4.8*** ms latency
<<peak rate, but ~10Mbp typical rate

*Qi Zheng, et al, "Parallelization Techniques for Implementing Trellis Algorithms on Graphics Processors", *ISCAS 2013*, Beijing

Outline

- Motivation
- LTE baseband kernels
- **Running LTE baseband on a mobile GPGPU**
  - Methodology
  - PHY layer
  - Turbo decoder
  - **Power and energy efficiency**
- Implications for future mobile GPGPUs
- Conclusion

Energy Efficiency



Higher frequency achieves better energy efficiency – *"Run to finish"*

Power Consumption

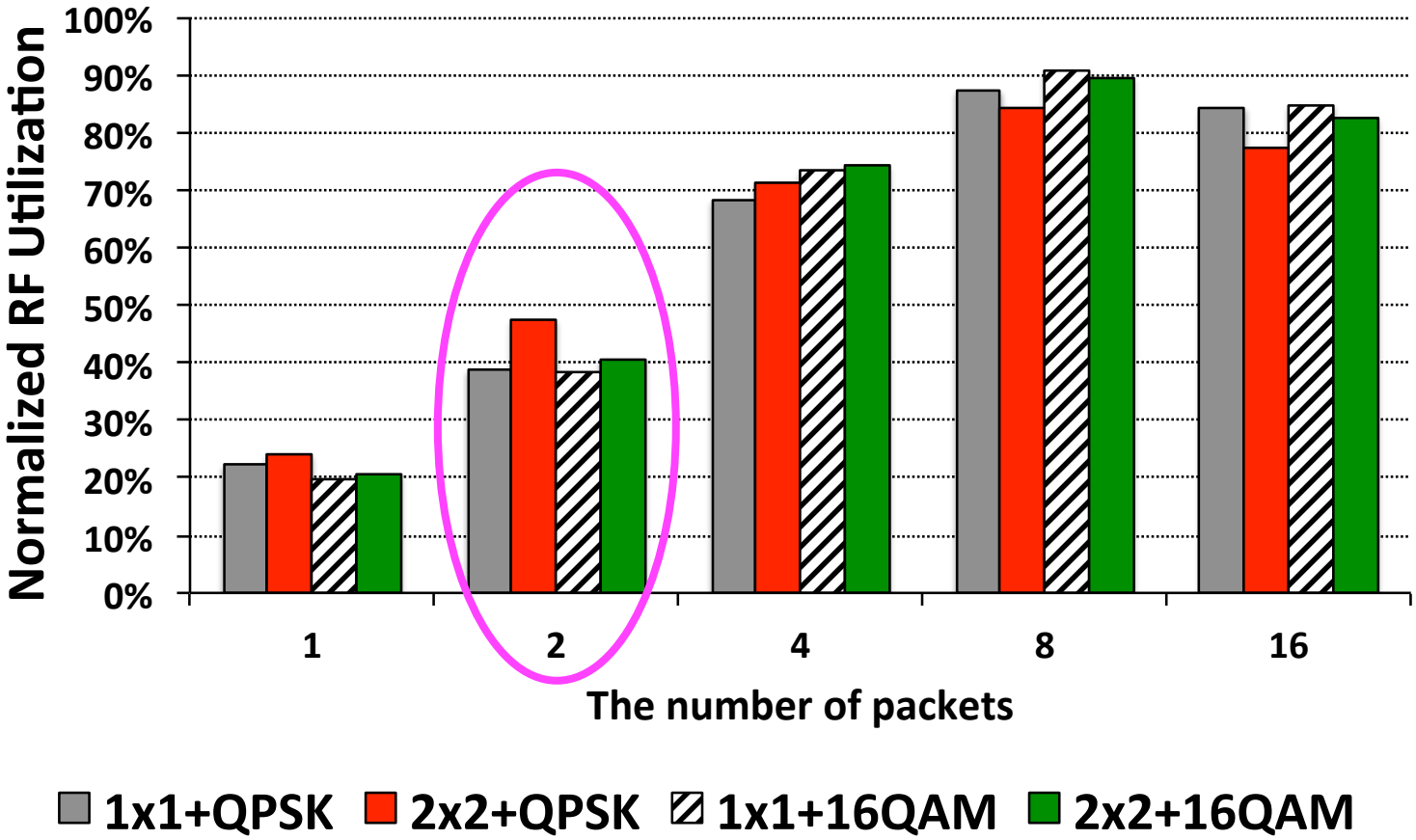
Processor	Standards	Details
ASIC (LeoCore)	WiMAX	70mW@ 70MHz
FPGA (MAGALI)	LTE	234mW@400MHz
DSP (Tomahawk)	LTE, WiMAX	904mW@170MHz
CPU (SoftLTE)	LTE	130W@2.66GHz
Mobile GPU	LTE	1390mW@852MHz

Outline

- Motivation
- LTE baseband kernels
- Running LTE baseband on a mobile GPGPU
- **Implications for future mobile GPGPUs**
- Conclusion

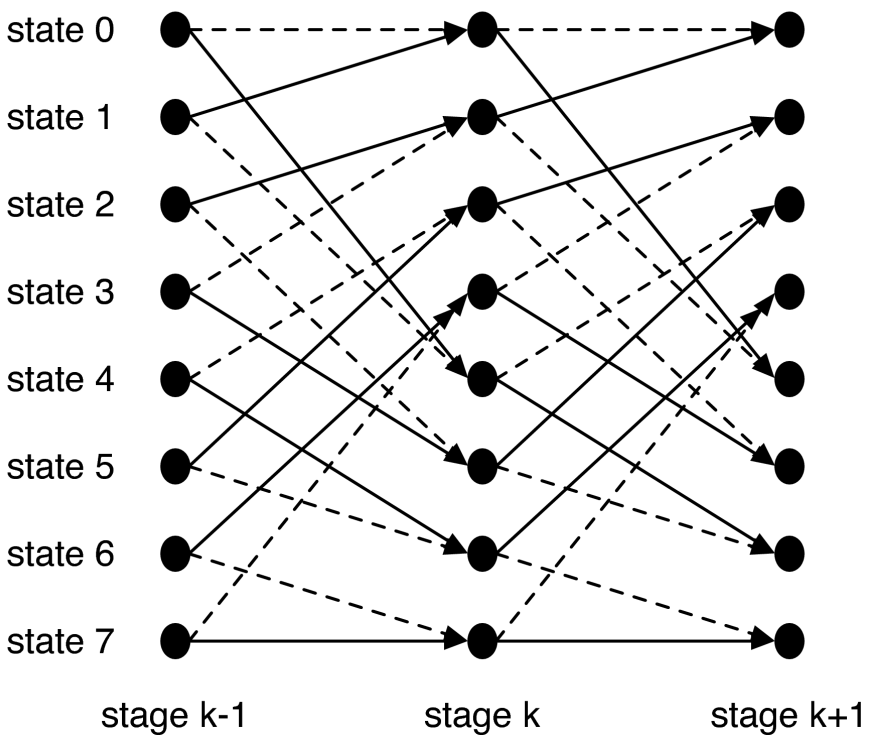
Reduce Energy from Unused Register

- Register file is underutilized running PHY layers
  - Due to max_thread/SM and max_thread_block/SM
- Power gate unused registers



GPU Support for Trellis Algorithm

- Trellis Algorithm
  - Turbo algorithm
  - Viterbi algorithm
  - Baum-Welch algorithm
  - Coding theory
  - Speech recognition
  - Data compression



- Architecture and ISA support
  - Similar to AES instruction set in Intel CPU

Outline

- Motivation
- LTE baseband kernels
- Running LTE baseband on a mobile GPGPU
- Implications for future mobile GPGPUs
- **Conclusion**

Conclusion

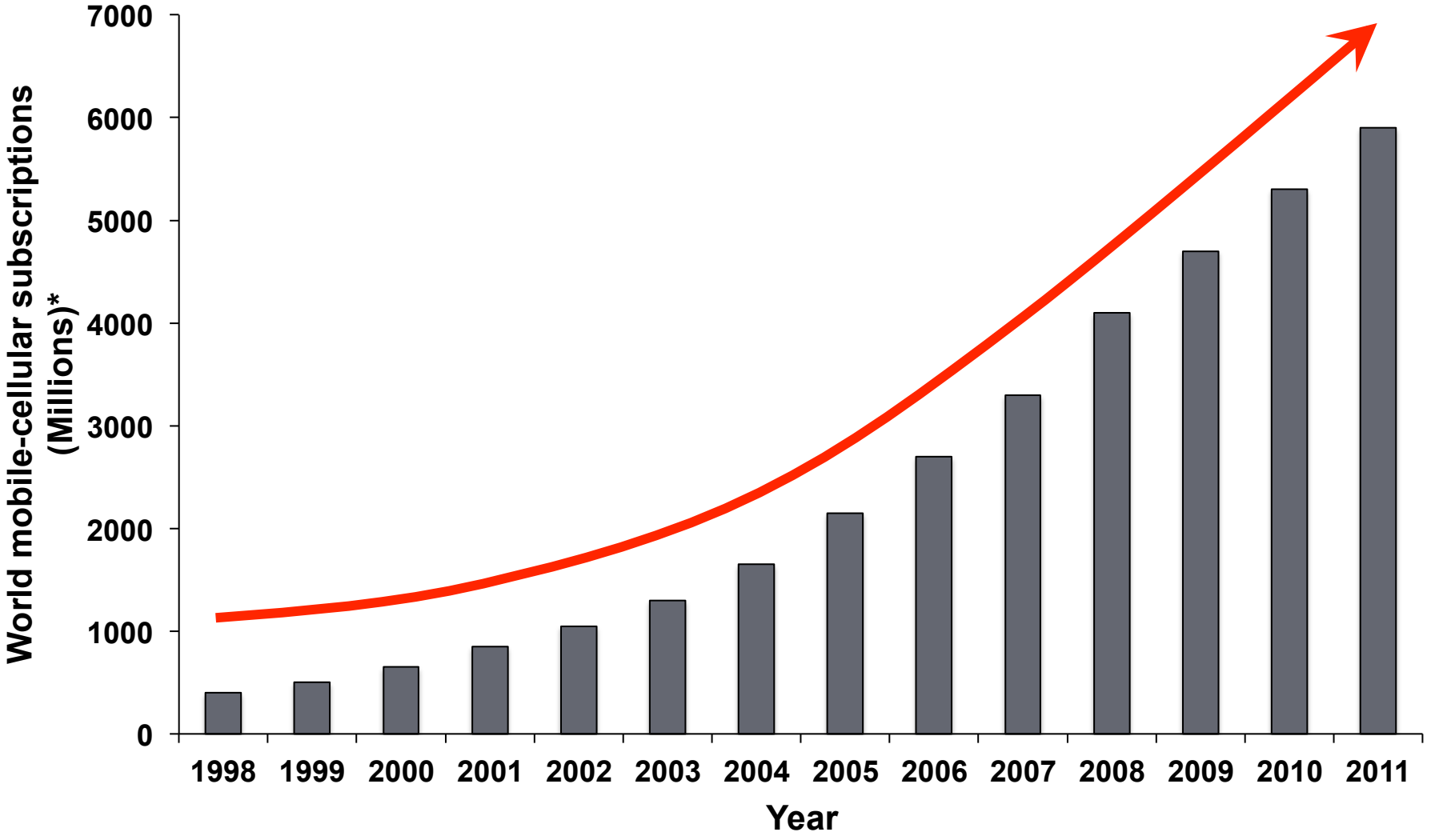
- Mobile GPUs can be used for wireless communication in a mobile device
  - Meet specification peaks for 3 out of 4 PHY configurations
  - Meet the typical rate for all PHY configurations
  - Turbo is close to the typical data rate
- Need special supports for the Turbo decoder in a mobile GPU
- Energy is high compared with application-specific solutions
  - But in the accepting range
  - Need further optimizations

Thanks!

Any questions?

Backup Slides

Quick Subscription Growth



*ITU report -- The *World in 2011: ICT Facts and Figures*

Our Contribution

- Parallel implementations of LTE baseband kernels on a mobile GPGPU
- Throughputs and latency of different configurations
- Energy efficiency of a mobile GPGPU

Parallelism in PHY Layer Kernels

Kernel	Parallelism	Number of threads
OFDM Modulation (FFT)	Antenna-level Symbol-level Algorithm-level	$N_{ant} \times N_{sym} \times N_{FFT}$
Channel Estimation	Antenna-level Subcarrier-level	$N_{ant} \times N_{sub}$
MIMO detector	Symbol-level Subcarrier-level	$N_{sym} \times N_{sub}$
Modulation demapper	Antenna-level Symbol-level Subcarrier-level Algorithm-level	$N_{ant} \times N_{sym} \times N_{sub} \times N_{Mod}$ $N_{ant} \times N_{sym} \times N_{sub} \times \log_2(N_{Mod})$
Descrambling	Antenna-level Symbol-level Subcarrier-level	$N_{ant} \times N_{sym} \times N_{sub}$

Parallelism in Turbo Decoder*

- Total number of threads

$$N_{thread} = N_{packet} \cdot N_{subblock} \cdot Thread_{trellis}$$

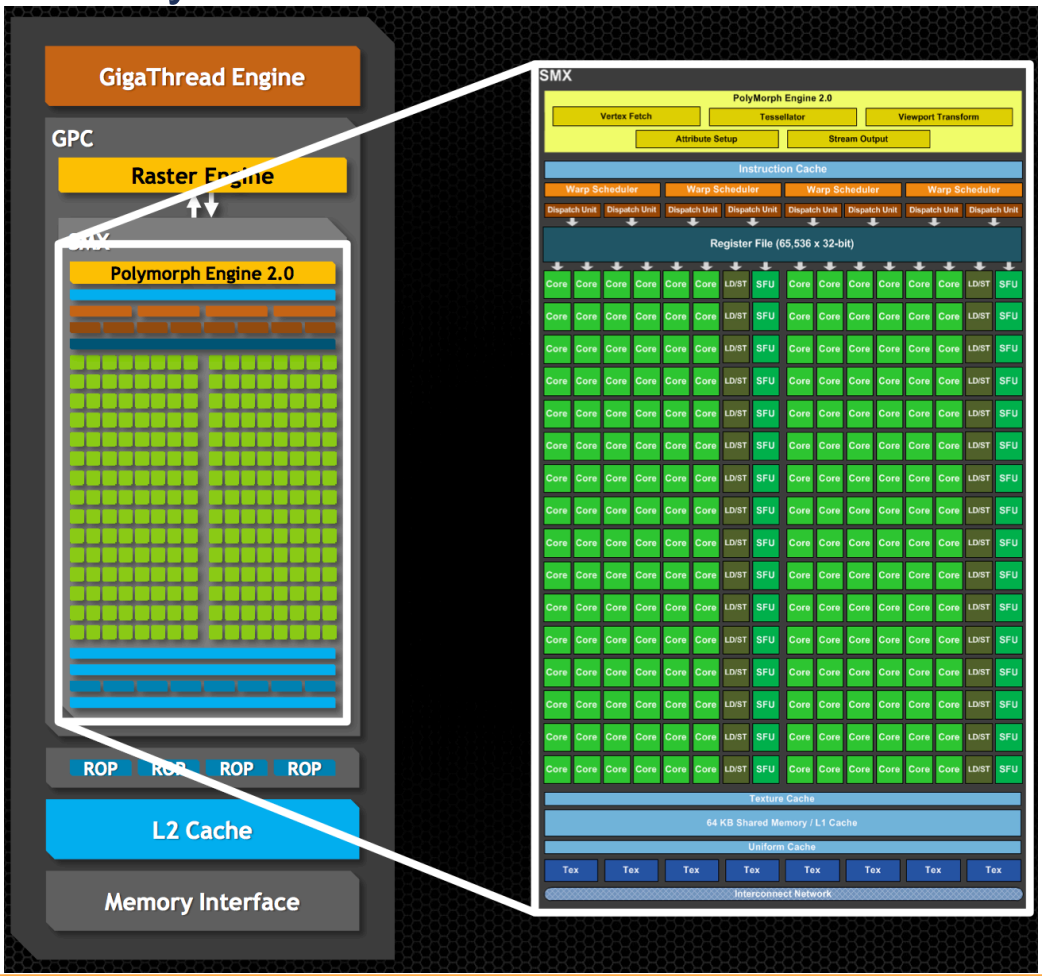
- Implementation performance tradeoff

Parallelism Scheme	Throughput	Latency	Bit Error Rate
Packet-level	Better	Worse	No Change
Subblock-level	Better	No Change	Worse
Trellis-level	Better	No Change	No Change
Subblock+NII	Worse	No Change	Better
Subblock+TS	Worse	No Change	Better

*Qi Zheng, et al, "Parallelization Techniques for Implementing Trellis Algorithms on Graphics Processors", *ISCAS 2013*, Beijing

Mobile GPU Architecture

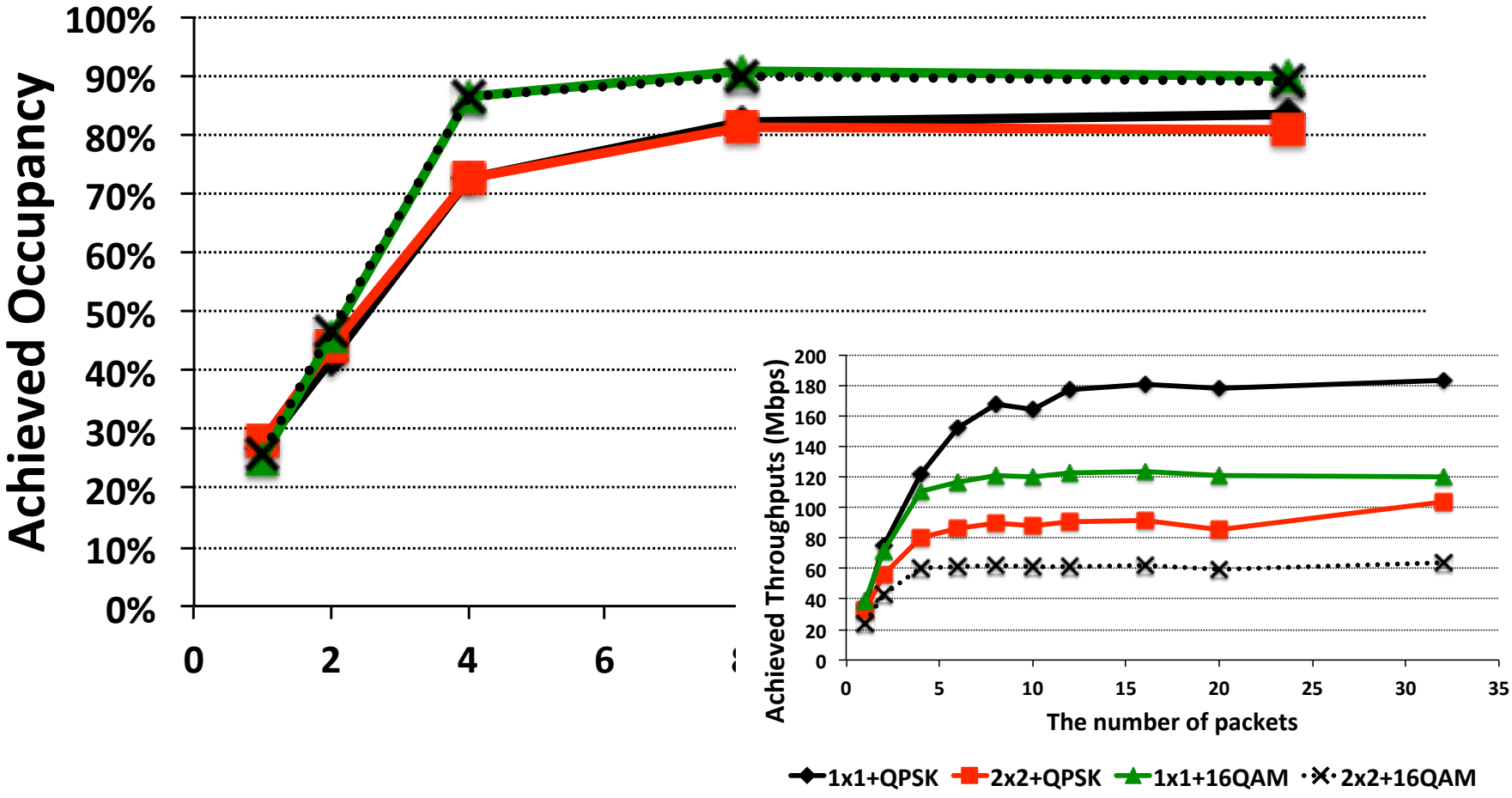
- GPU on NVIDIA Jetson Tegra K1 board
  - Kepler architecture – 192 CUDA cores@852 MHz
  - 64KB L1 memory / 128KB L2 cache / Share 2GB DDR3L with CPU



Performance Metrics

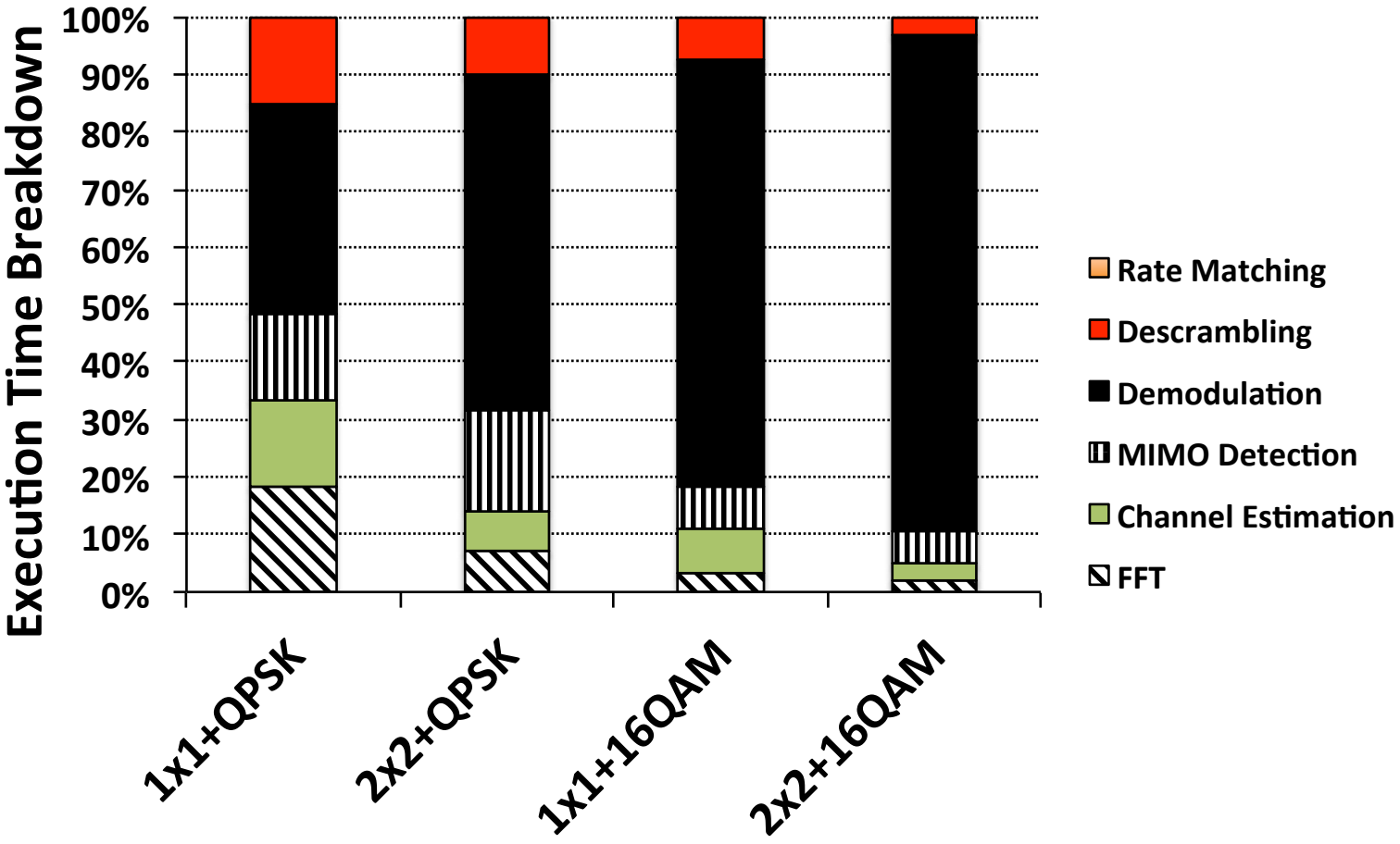
- Collected stats for a variety of important events
- CUDA Event API
  - Throughput
  - Latency
- NVIDIA Profiler
  - Achieved occupancy
  - Memory utilization
  - Dynamic instruction count

GPU Occupancy



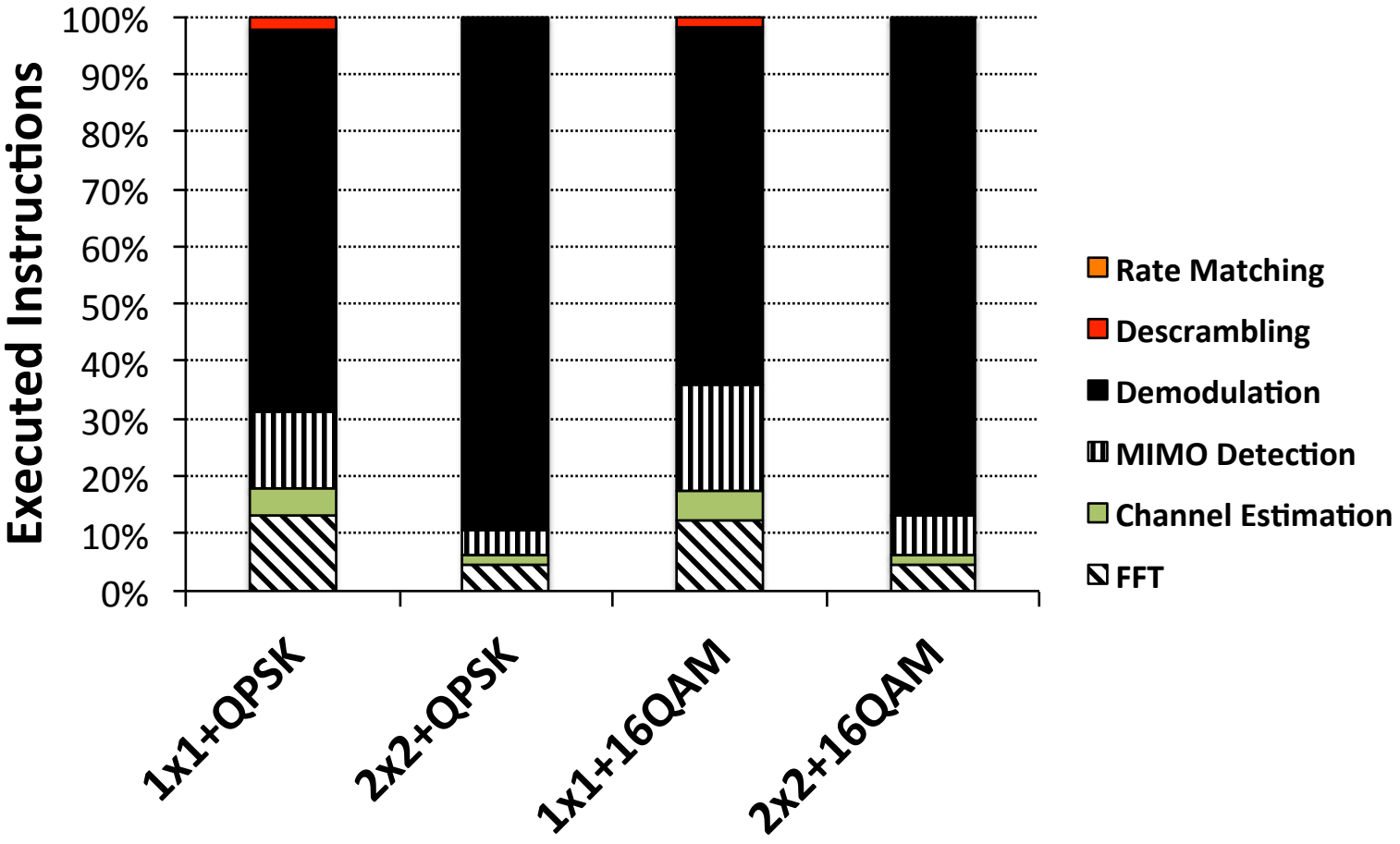
Packet batching increases throughput…
which saturates at 8 packets due to full GPU utilization

Hotspot Analysis



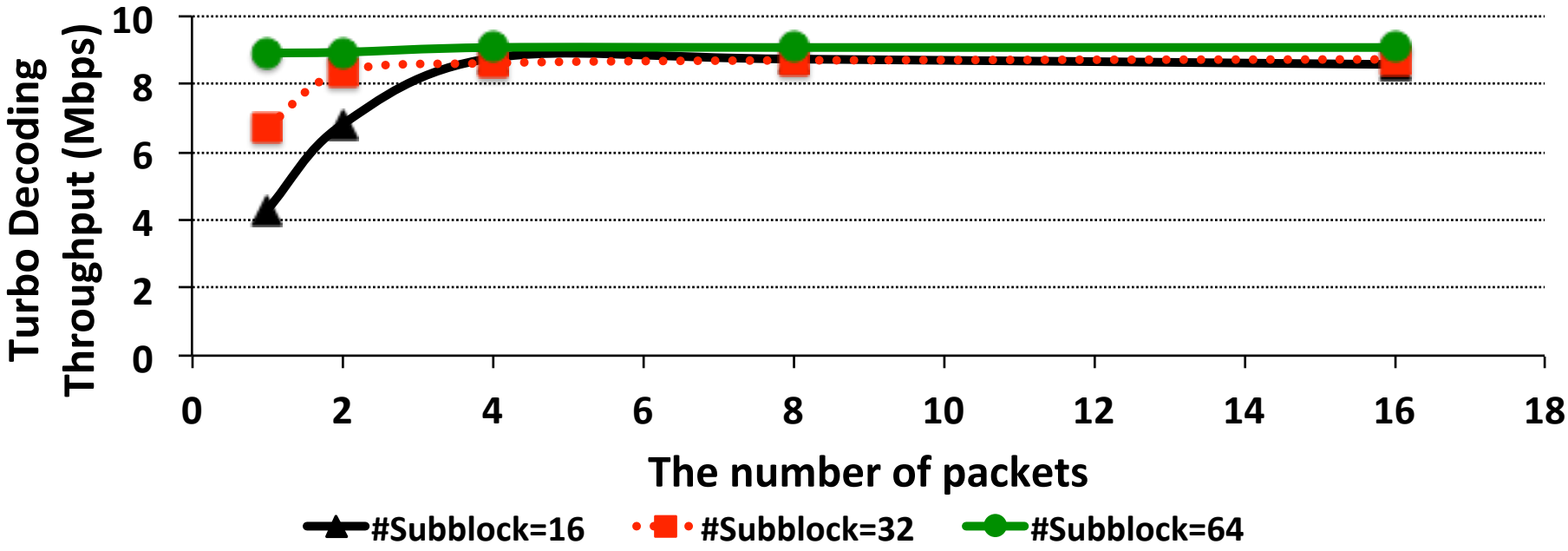
Demodulation dominates due to its heavy computation

Hotspot Analysis -- #instructions



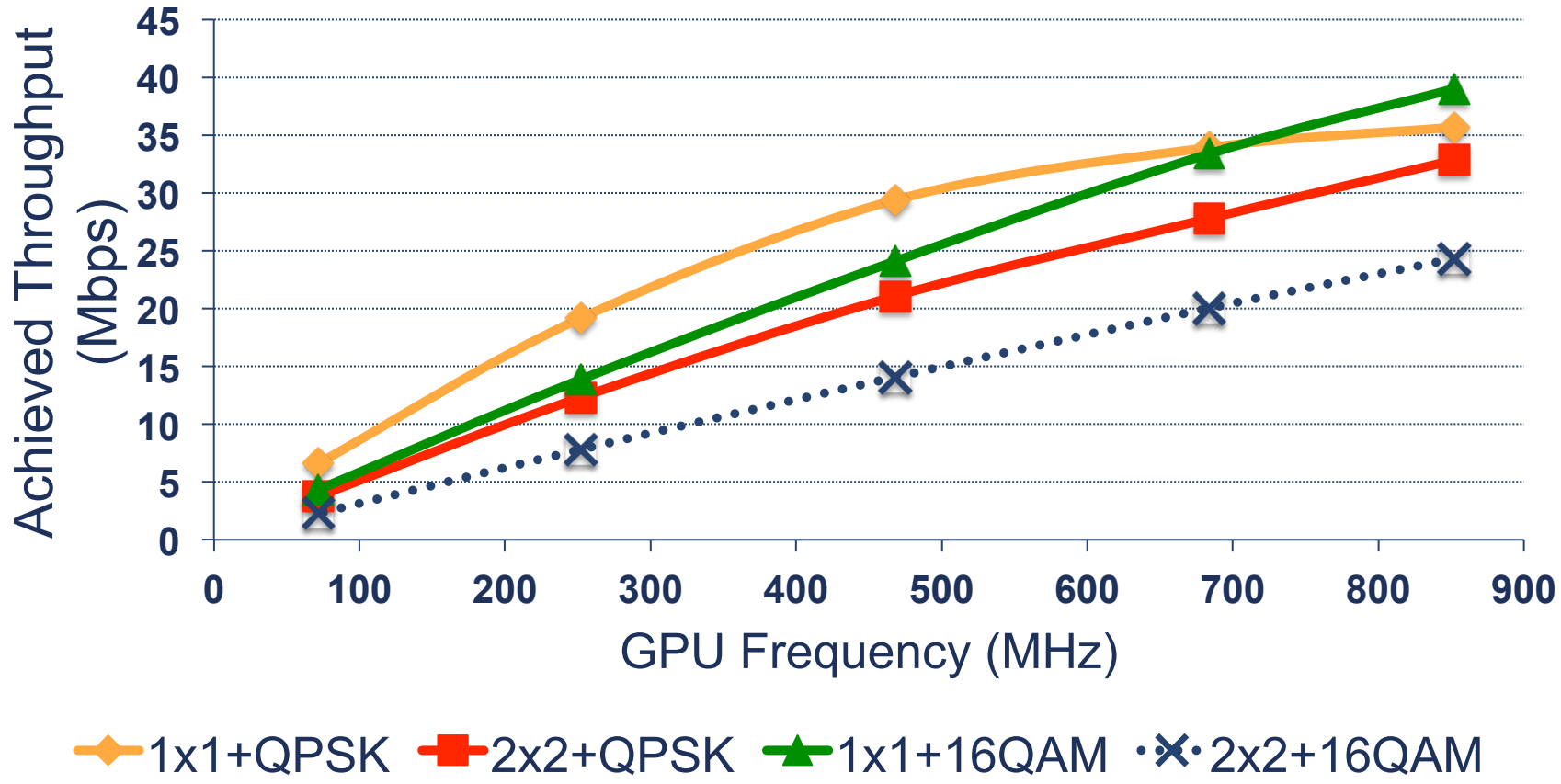
Demodulation dominates due to its heavy computation

Turbo Throughput



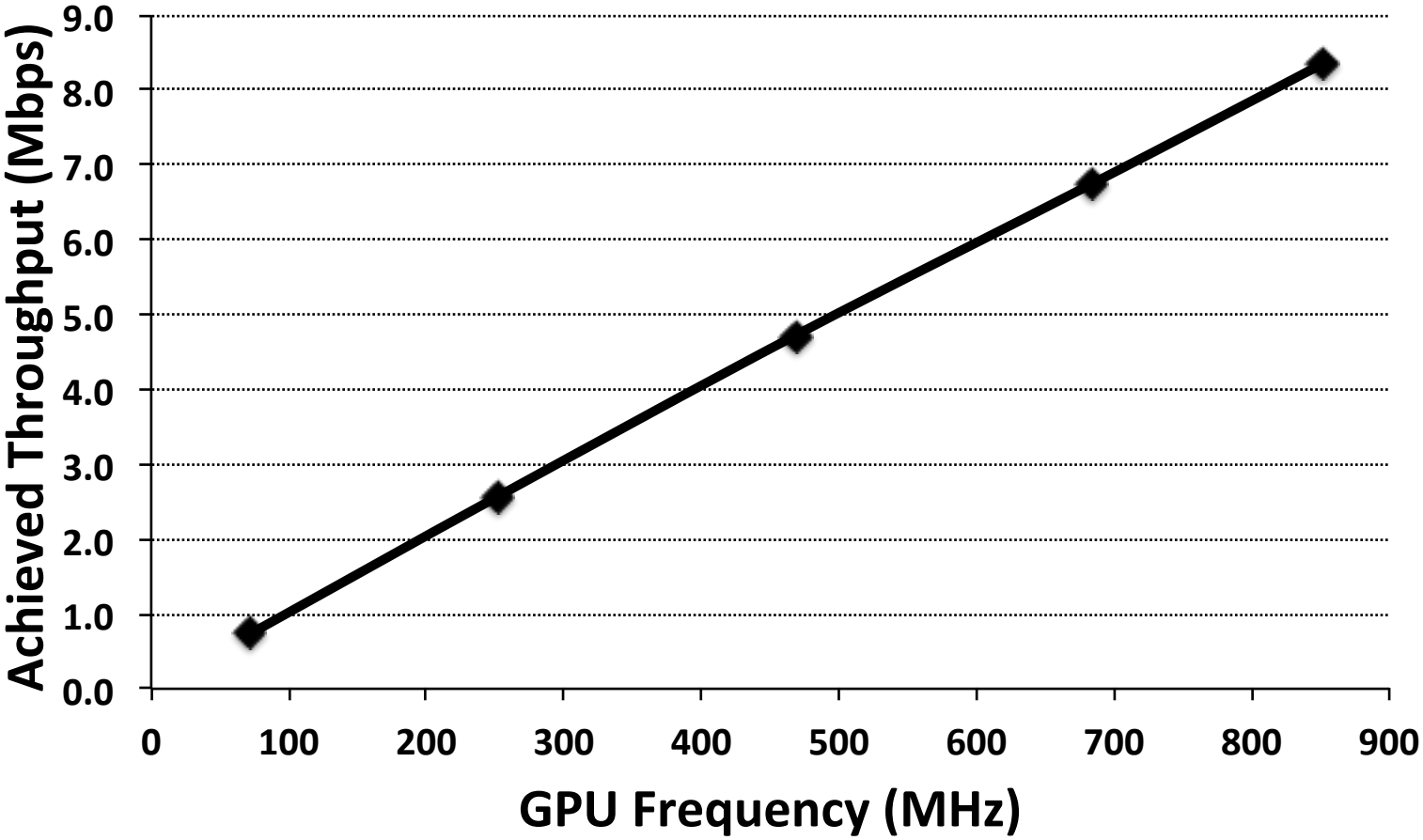
Throughput increases with the number of subblocks and packets, and starts saturating at 64 subblocks

Frequency Scaling -- PHY



Throughputs have almost linear scaling with the frequency

Frequency Scaling -- Turbo



Throughputs have almost linear scaling with the frequency