Performance Study of a GPU in Real-Time Applications for HEP Experiments

Wesley Ketchum[1],

Silvia Amerio[2], Denis Bastieri[2,3], Matteo Bauce[2,3], Pierluigi Catastini[4], Kristian Hahn[4], Young-Kee Kim[1,4], Tiehui Liu[4], Donatella Lucchesi[2,3], and Giorgio Urso[5]

(1) University of Chicago, (2) INFN Padova,

(3) University of Padova, (4) FNAL, (5) ORMA Software

Introduction

- Research project interested in basic R&D for new trigger techniques
- Use resources available at CDF trigger test stand
  - Hardware
  - Testing software
  - People
- Outline
  - Why we're interested in GPUs
  - Our experimental setup
  - Current measurements
  - Looking ahead…

Motivation

- Power of GPUs has increased rapidly due to demands of 3D graphics
  - Highly parallelized architecture
  - High memory bandwidth
- Many applications of GPUs outside of imaging
  - Commercially available → cheaper than dedicated hardware
  - Application programming interfaces like nVidia's CUDA C ease development of software for new applications

Photograph of GTX 285 GPU, courtesy of nVidia.

- ***Are GPUs suitable for low-latency environments, like a HEP trigger?***

GPU vs CPU Computation

CPU (Intel Core i7-930)

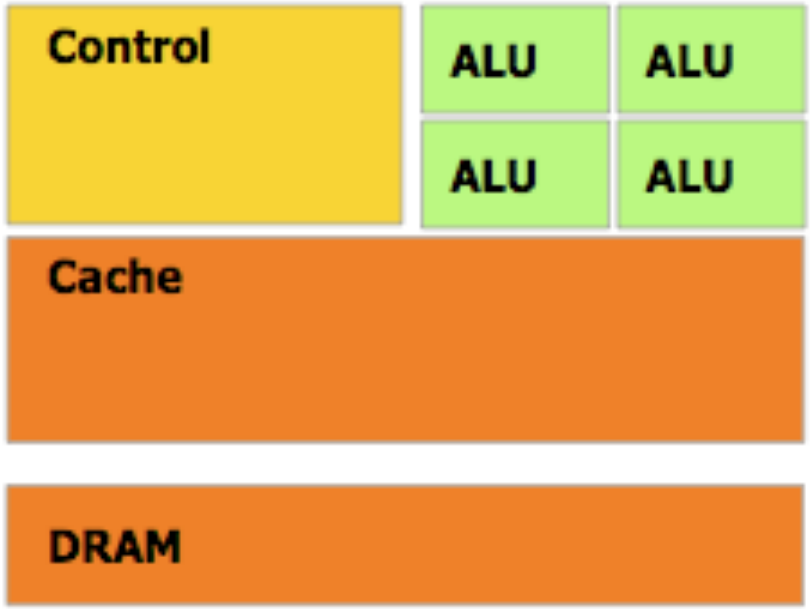
- Limited number of simultaneous calculations possible
  - 1 microprocessor
  - 4 cores
  - 8 threads
- Large cache size
  - 8 MB

GPU (nVidia GeForce GTX 285)

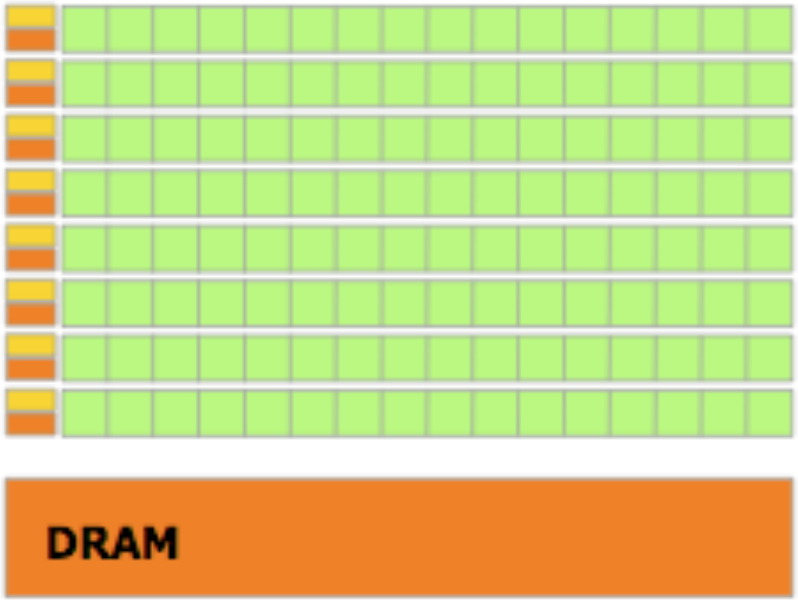
- Designed for running many instances of same routine simultaneously
  - 30 microprocessors
  - 240 cores
  - 1024 x 30 threads (max)
- Small cache size
  - 8 kB / microprocessor

GPU vs CPU Computation

CPU (Intel Core i7-930)



GPU (nVidia GeForce GTX 285)



From nVidia CUDA C Programming Guide (v 3.2)

GPU vs CPU Computation

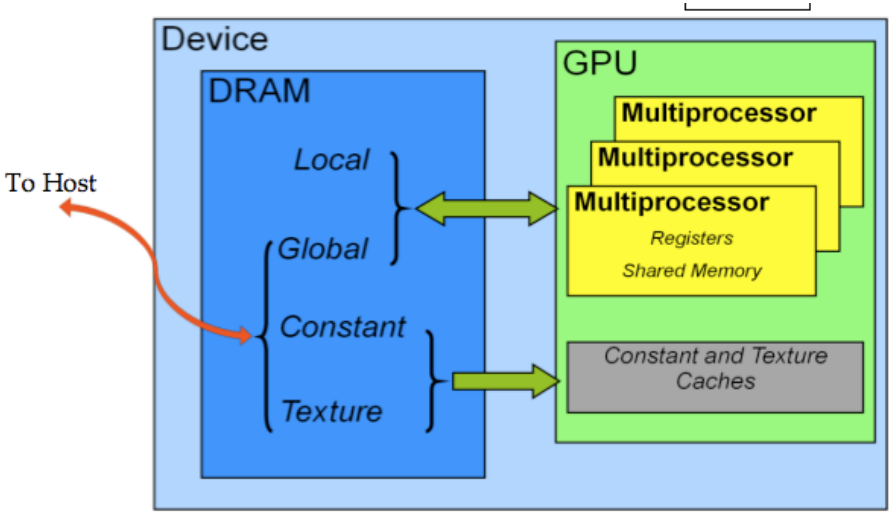
CPU (Intel Core i7-930)

- Limited number of simultaneous calculations possible
  - 1 microprocessor
  - 4 cores
  - 8 threads
- Large cache size
  - 8 MB
- Sits directly on motherboard
  - Latency scale set by number/ speed of operations

GPU (nVidia GeForce GTX 285)

- Designed for running many instances of same routine simultaneously
  - 30 microprocessors
  - 240 cores
  - 1024 x 30 threads (max)
- Small cache size
  - 8 kB / microprocessor
- Communicates with CPU through PCIe bus
  - Latency scale set by host (CPU) ↔ device (GPU) communication

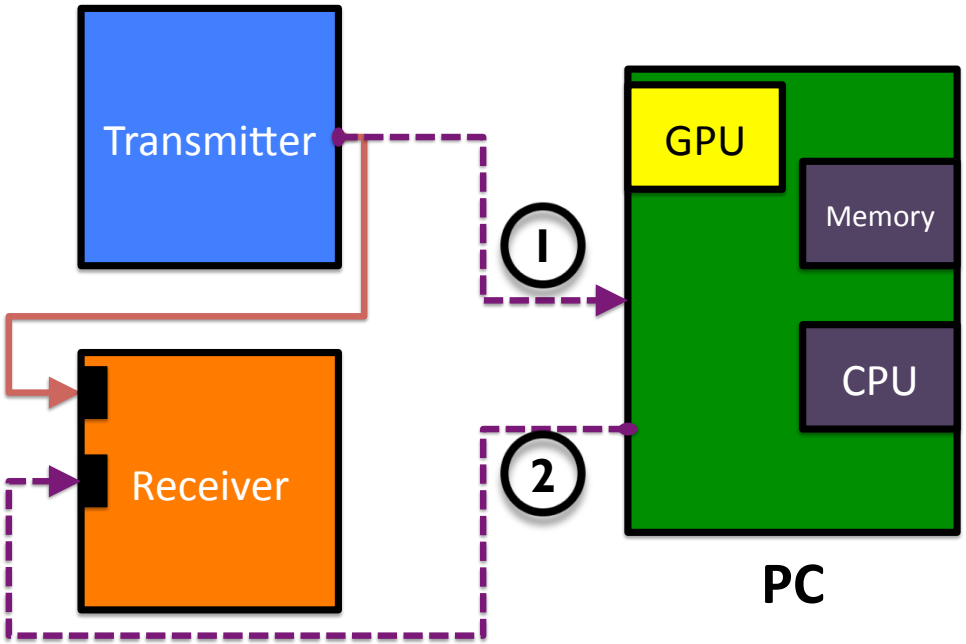
GPU Memory Structure



From nVidia CUDA C Best Practices Guide (v 4.0)

- Various memory locations for storing/accessing data
  - Global Memory
    - Most available space
    - Read/Write
    - Slow access
  - Constant/Texture Memory
    - Smaller storage space
    - Read Only
    - Cacheable on multiprocessors (faster access)
  - Registers/Shared Memory
    - Limited storage space
    - Read/Write
    - Fast access for individual threads for thread blocks

Experimental Setup: Data Flow



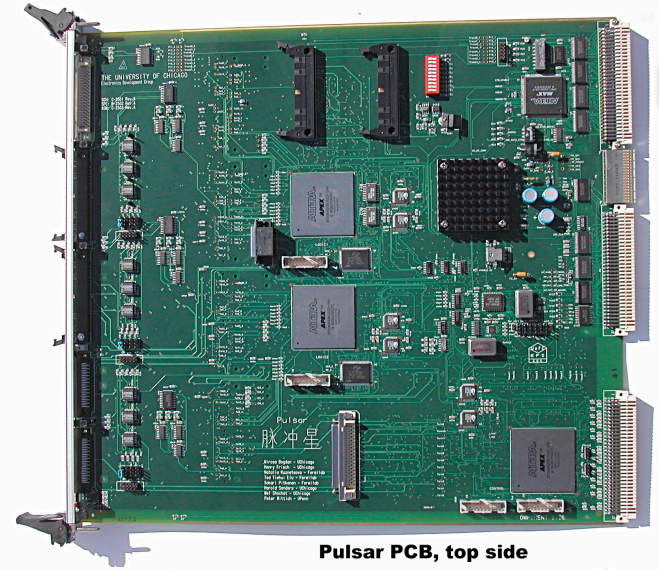
Steps in PC

- Receive input data
- Copy input to GPU
- Perform calculations
- Copy results from GPU
- Send output

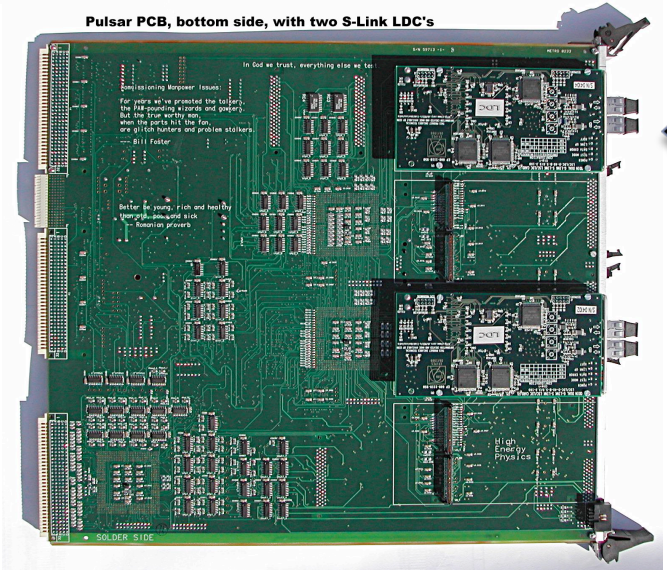
Goal: Measure total time for performing an HEP trigger algorithm from input going into the PC (t_1) and the output leaving the PC (t_2) and determine latency ($t_2 - t_1$)

Input/Output: PULSARS

- PULSAR (PULSer And Recorder) boards used in CDF Level 2 trigger system
  - Highly configurable
    - Transmit/receive CERN S-LINK
- Perform studies at CDF L2 Test Stand
  - Measure arrival time of data packets very well
  - PC running in real-time trigger environment



Pulsar PCB, top side



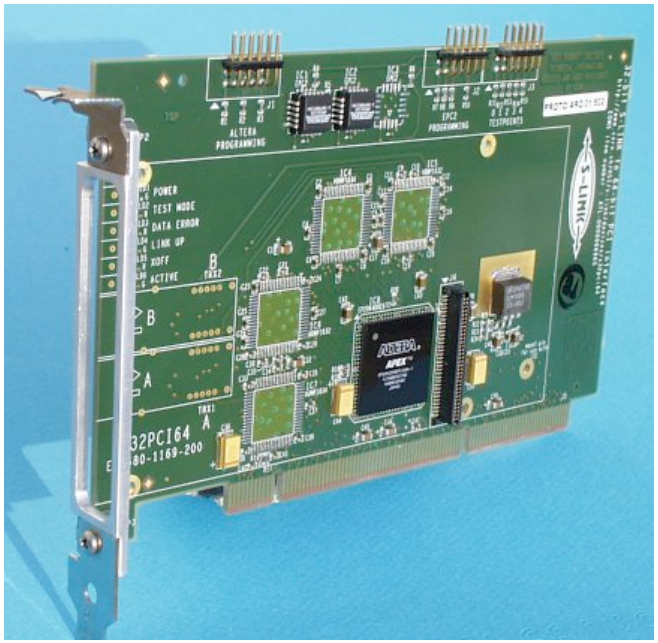
Pulsar PCB, bottom side, with two S-Link LDC's

Connects to VME backplane

Receives S-LINK packets

Input/Output: S-LINK PCI Cards

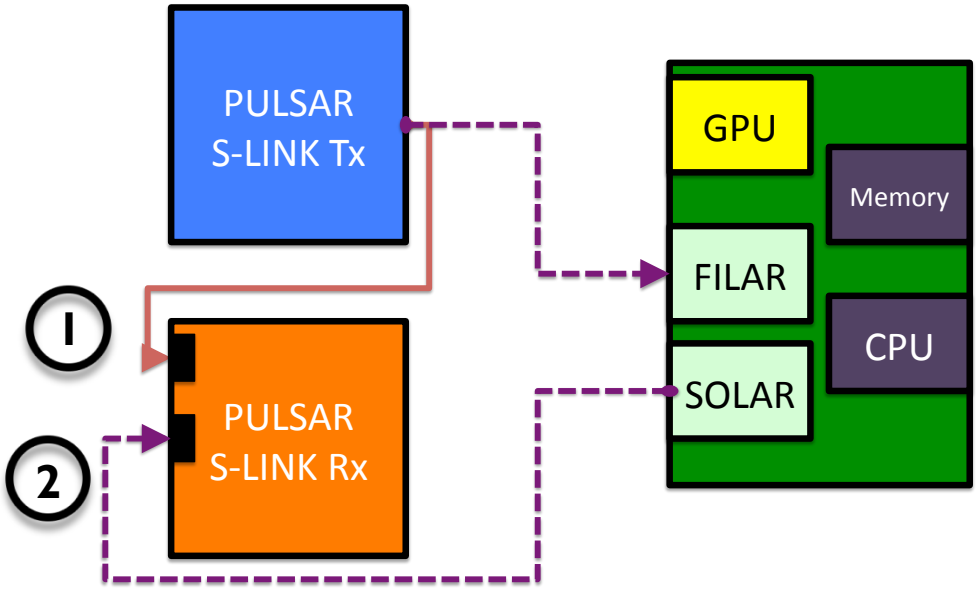
FILAR (above) and SOLAR (left, without S-LINK mezzanine attached).



From http://hsi.web.cern.ch/HSI/s-link/devices

- S-LINK data received/sent on special PCI cards
  - FILAR (Four Input Links for Atlas Readout)
  - SOLAR (Single Output …)
- FILAR/SOLAR cards used in current CDF L2 system
  - Inherit drivers/operation code from L2 upgrade effort

Experimental Setup: Data Flow



Steps in PC

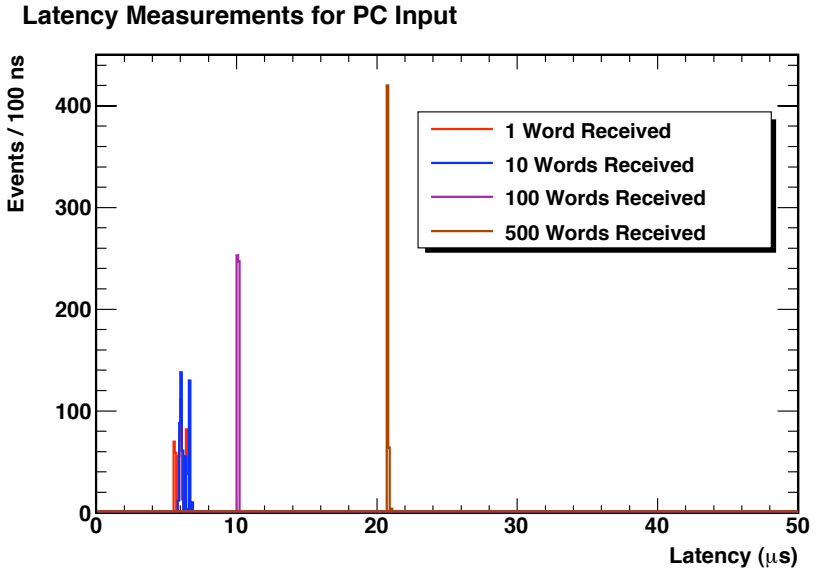
- Receive input data
- Copy input to GPU
- Perform calculations
- Copy results from GPU
- Send output

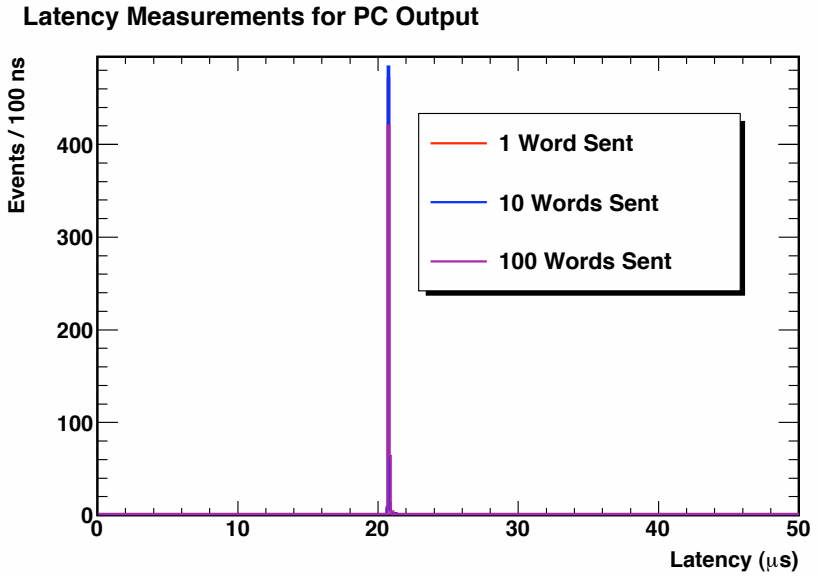
PC↔PULSAR Communication

- PULSAR sends hit combinations to PC
  - Default: 500 S-LINK words → 20.5 μ s latency
- PC sends back some of results to PULSAR
  - Default: 100 S-LINK words
    - doesn't contribute much to latency

Steps in PC

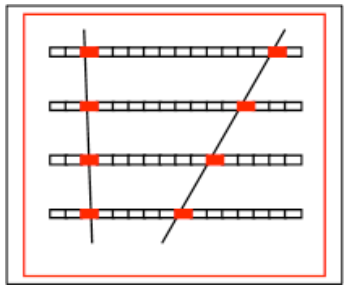
- Receive packets on FILAR
- *Copy input to GPU*
- *Perform calculations*
- *Copy results from GPU*
- Send output to PULSAR



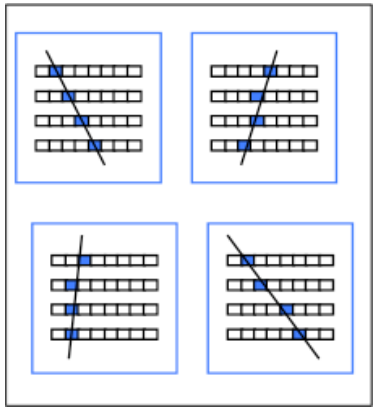


The Computation: Linearized Track Fitting

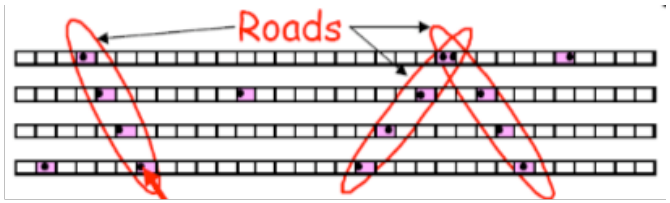
- Want to run algorithm that would be used in HEP trigger
- CDF Silicon Vertex Trigger (SVT) finds displaced vertices at L2
  - Pattern Recognition to form hit combinations (roads)



Event Hits



Compare to Pattern Bank



- Perform track-fitting inside roads using simple scalar product

$$p_i = \vec{f}_i \cdot \vec{x} + q_i$$

Known constants. Precalculated and ***stored in memory***

track parameters (output)

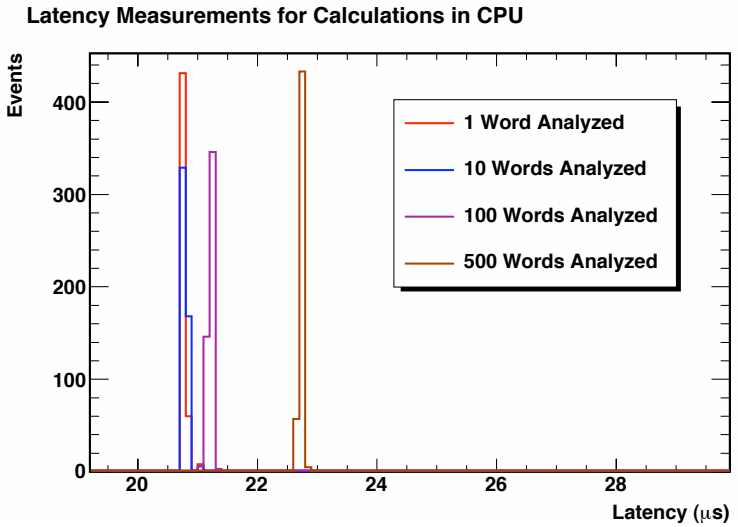
track coordinates (input hit information)

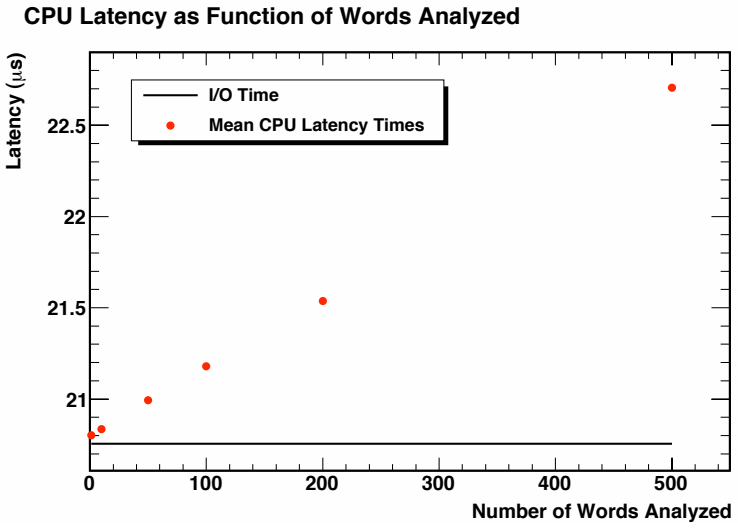
Calculations in CPU Only

- Run track-fitting algorithm to "fit" fixed number of tracks
  - Fixed input and output word lengths
  - Fit 1 track (= 1 word) at a time
- Small spread in CPU latency times
- Mean latency increases linearly

Summary of Data Flow

- Receive packets on FILAR
- *Copy input to GPU*
- Perform calculations (CPU)
- *Copy results from GPU*
- Send output to PULSAR



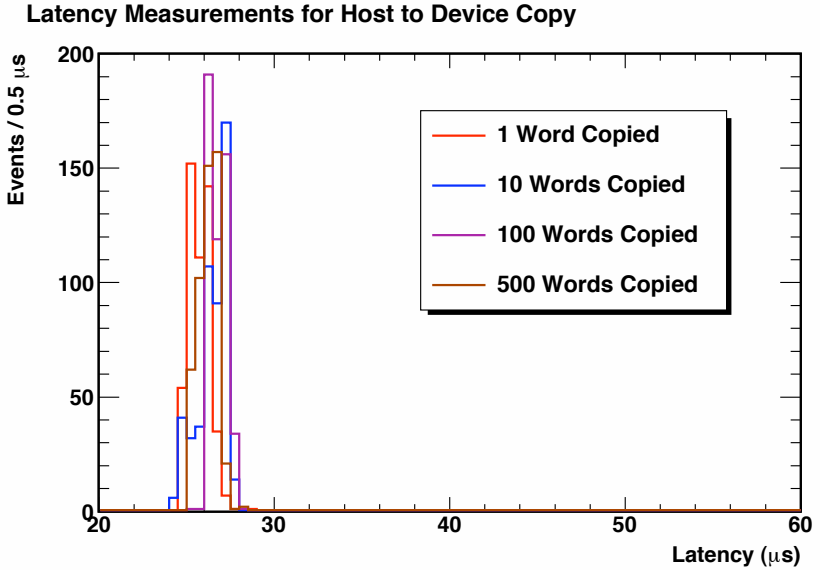


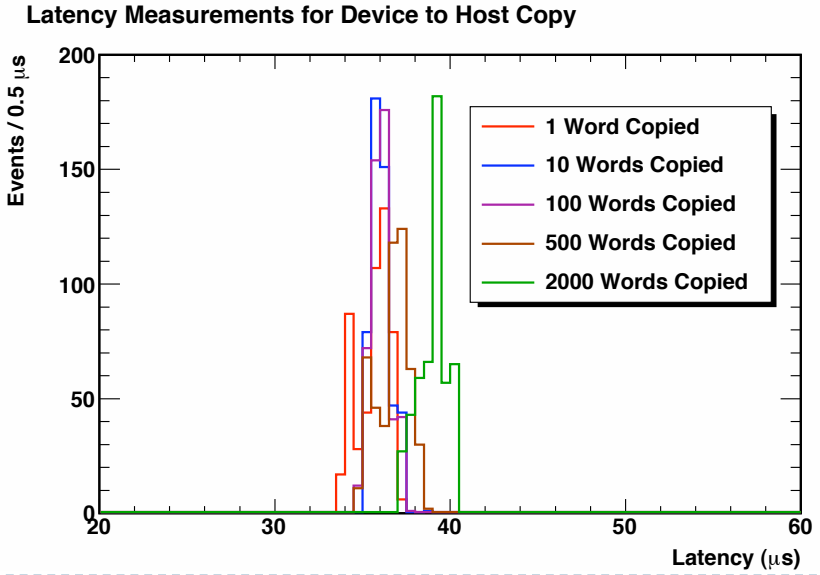
CPU↔GPU (Host↔Device) Communication

- Copy input words to GPU global memory
  - Default: 500 words → 6 μ s latency
- Copy results from GPU back to CPU
  - Default: 2000 words (4 output words for each input word) → 19 μ s

Summary of Data Flow

- Receive packets on FILAR
- Copy input to GPU
- *Perform calculations*
- Copy results from GPU
- Send output to PULSAR



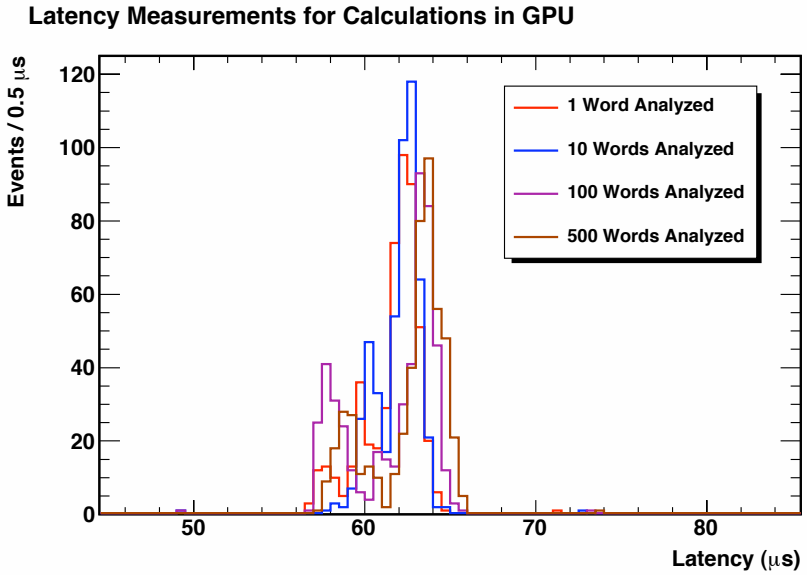


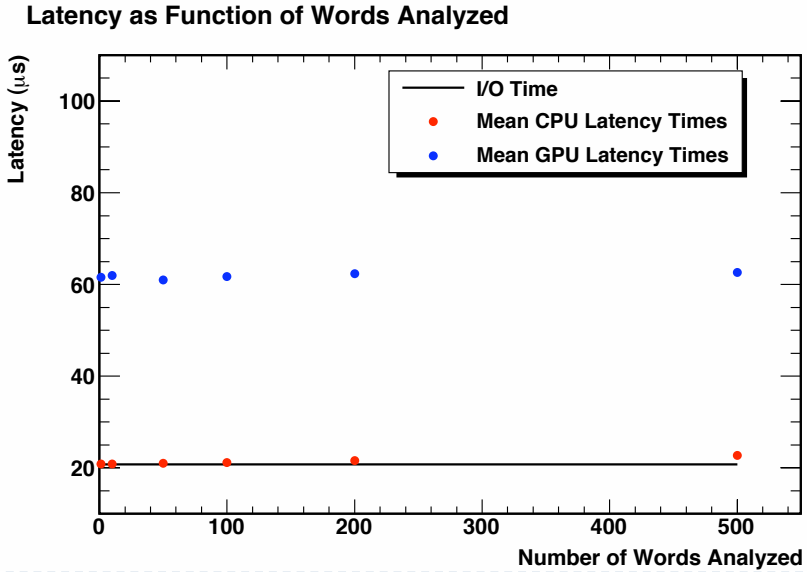
Calculations in GPU

- Run track-fitting algorithm: one track fit per thread
  - Amount of memory transfer between CPU and GPU held constant
- As compared to CPU…
  - Latencies much longer in GPU (~60 μ s total)
  - Spread of latencies much larger in GPU

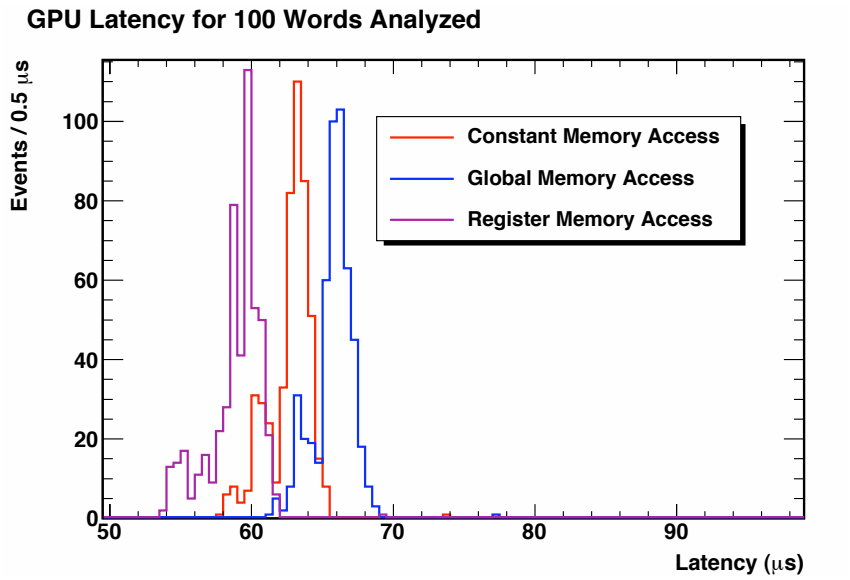
Summary of Data Flow

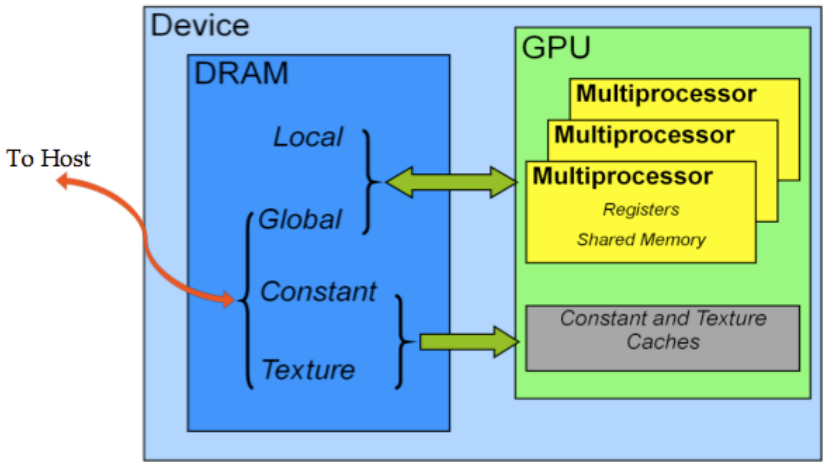
- Receive packets on FILAR
- Copy input to GPU
- Perform calculations
- Copy results from GPU
- Send output to PULSAR





Varying GPU Memory Lookup





From nVidia CUDA C Best Practices Guide (v 4.0)

- Algorithm accesses pre-defined constants for track fitting
  - $p_i = \vec{f}_i \cdot \vec{x} + q_i$
- Location in memory affects latency
- Significant dependence of latency on handling of ***memory lookup***
  - Differences ~ 10 μ s between register and global memory
- Good management → optimized performance

Future Measurements

- Further testing of runtime properties of GPU
  - Optimal thread management in GPU
  - Strategies for addressing long latency of host ↔ device communication
  - Memory access strategies within GPU
  - **ALL within context of real-time trigger system**
- Capable of testing more complex code:
  - Construct silicon hit combinations inside GPU
  - Perform calorimeter tower cluster for jet triggers
  - Directly compare performance of full trigger algorithms to current CDF L2 benchmarks

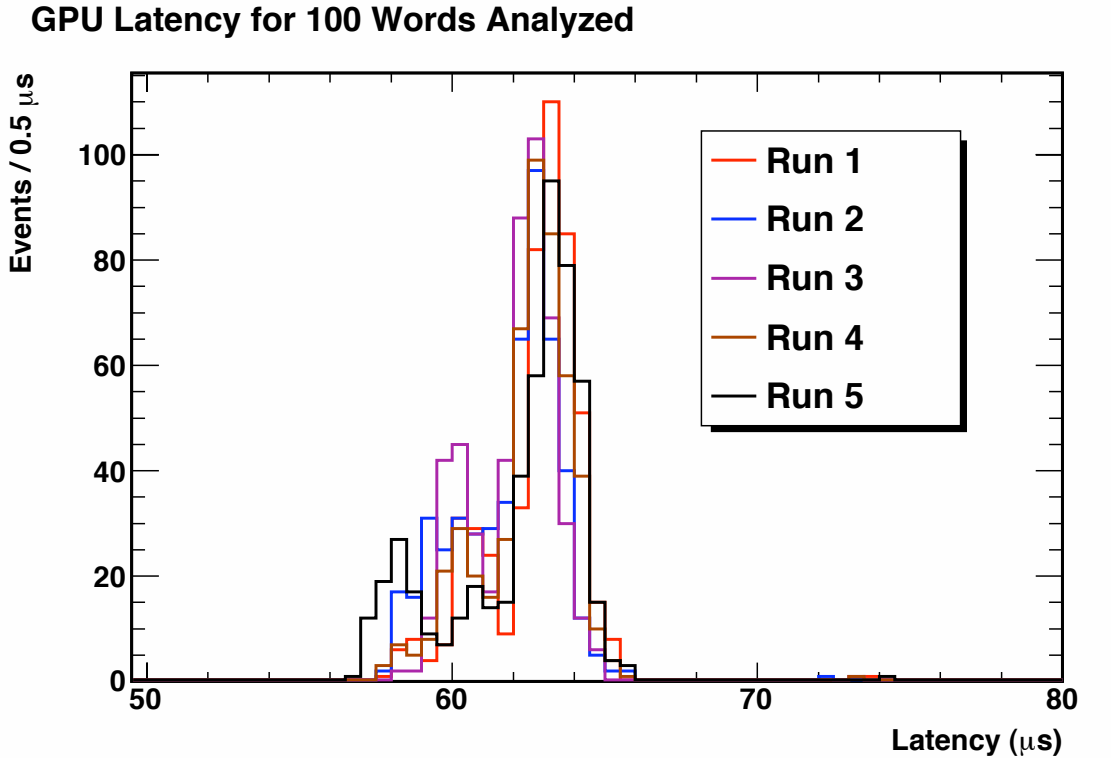
Conclusions

- GPUs promising idea for future HEP trigger applications
  - Designed for running parallel algorithms with high memory bandwidth
  - Familiar software development
  - Commercial product in a consumer-driven market
- Still, some limitations to be investigated and understood
  - Slow latency for host↔device communication
  - Sensitivity to memory access requires careful optimization
- CDF L2 test stand hosts detailed performance studies in a real-time trigger environment
  - Established some base line performance marks
  - More detailed studies underway!

Backup Slides

Typical Spread in GPU

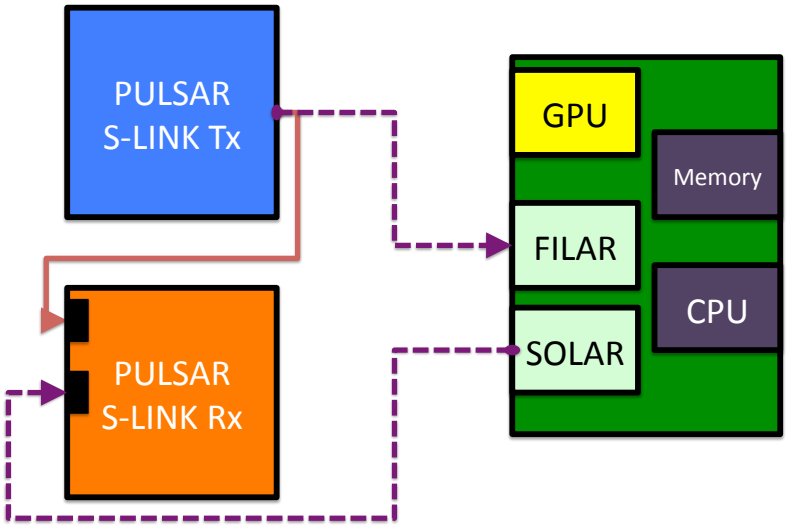
- Mean of GPU latency measurements can vary from run to run
  - Means vary by ~ 0.3 μ s



Outline of Results

- IO Time for Receiving and Sending Signals
  - As function of Number of Input Words
  - As function of Number of Output Words
- Latency for Host→Device and Device→Host Copying
  - As function of Input/Output Words
- Latency for CPU measurements
  - Varying number of calculations
- Latency for GPU measurements
  - For constant number of calculations
  - Varying number of calculations
  - Varying type of memory access

Experimental Setup: Data Flow



- Inputs loaded into S-LINK transmitter PULSAR
  - Use each S-LINK word to represent one "hit combination" set
- Send patterns to PC and S-LINK receiver directly
- In the PC:
  - Receive packets on FILAR
  - Copy input to GPU
  - Perform calculations
  - Copy output from GPU
  - Send output to Pulsar on SOLAR
- S-LINK Receiver PULSAR measures time of incoming packets for latency