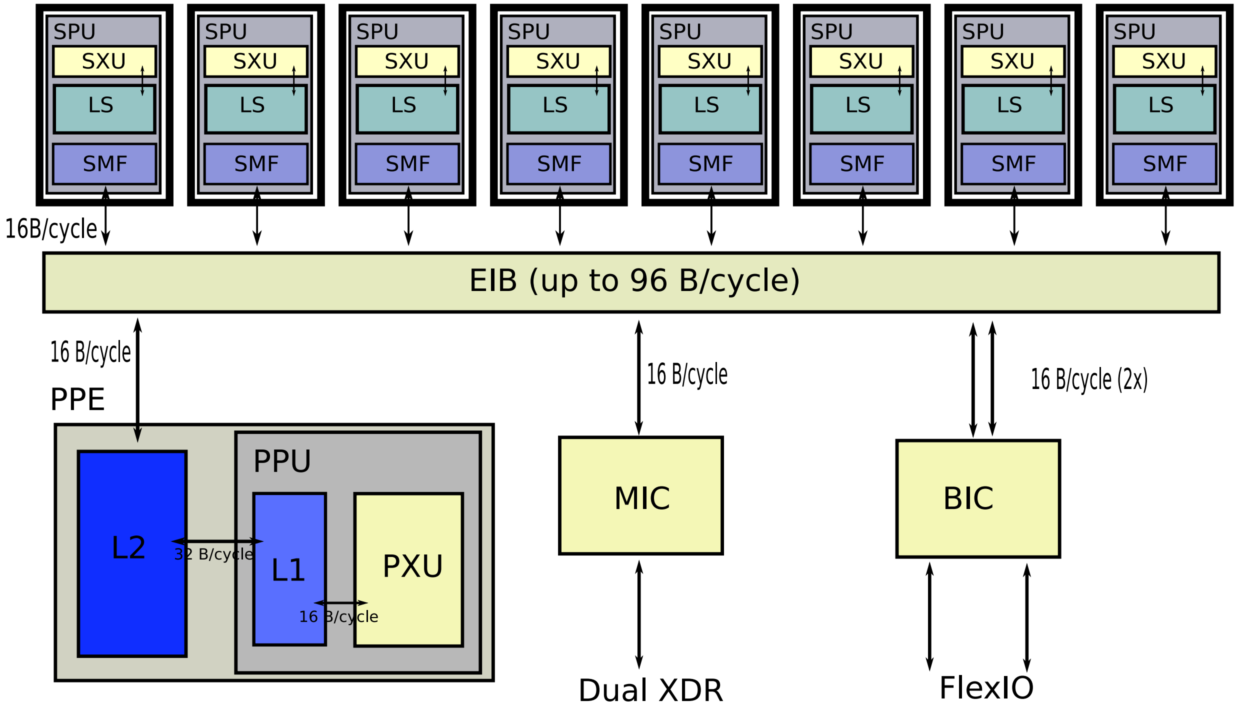
The DUP System

Christian Grothoff
christian@grothoff.org

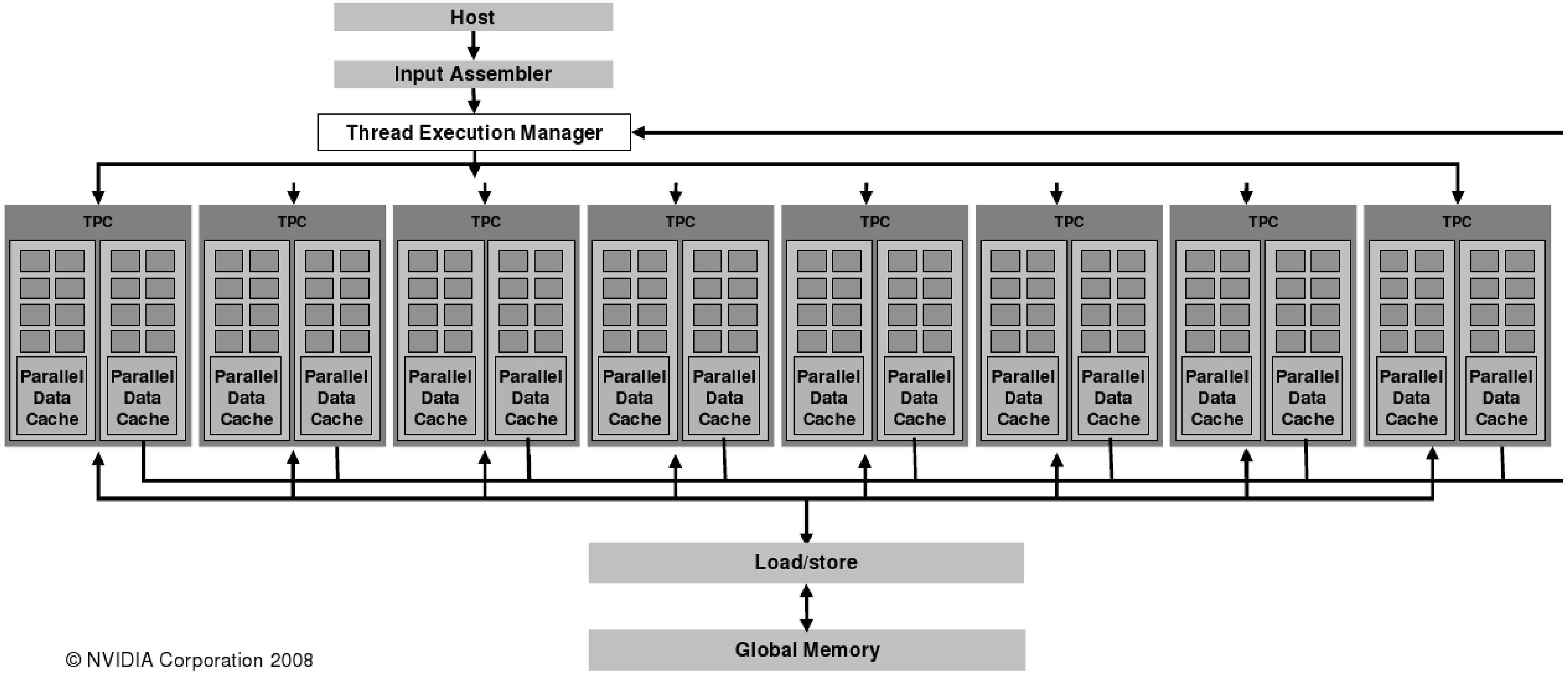
joint work with
Chris GauthierDickey and Matthew Rutherford

The Problem



64-bit Power Architecture with VMX

The Problem


Host
Input Assembler
Thread Execution Manager
TPC
Parallel Data Cache
Load/store
Global Memory
© NVIDIA Corporation 2008

How Much Faster?[1]

- Visualization: 146x
- Turbulence simulation: 17x
- Nbody simulation: 100x
- Molecular dynamics: 24x
- Gene sequence matching: 30x

[1] According to http://www.nvidia.com/docs/IO/47904/VolumeI.pdf

The Problem: Developing Parallel Stream Applications

- Most developers (only) know how to write sequential code
- Parallel programing is error-prone (data races, deadlocks)
- High-performance parallel programming is really hard
- With GPUs for $4,000, we could have 2,600 cores...

⇒ Developers more expensive than hardware

A Blast from the Past: CMS Pipelines

- Like UNIX pipes in use
- Sligthly different syntax
- NEW: multistream pipelines

CMS Pipelines

```
Pipe < INPUT FILE A % input is a stage!
| drop 4            % like ‘‘eat 4’’
| locate 5.1 /4/    % grep 4 in colum 5
| sort 34-36        % sort by colums 34-36
| > OUTPUT FILE A   % output is a stage!
```

CMS Pipeline Terminology

- Stage – Program that accomplishes a specific task
- Stage Separator – |
- Stream – flow of data into and out of a stage
- Device Driver – stage that interfaces with the environment
- Filter – processes data without interfacing with environment

Common Filters

- locate, find, nlocate, nfind – select records with specified target
- between, inside, outside, ninside – select records between specified targets
- take, drop – select records by counter
- unique, sort unique – select unique records
- sort – sorting
- combine, overlay – combine records
- duplicate – duplicate records

Common Filters

- specs, change, chop, strip, pad – manipulate record data
- block, deblock, split, spill, join, joincont – block and unblock records

Multistream Pipelines

- Multistream pipelines are pipelines that contains stages that have multiple input or output streams

Multistream pipelines introduce a new potential problem: pipeline stalls.

Writing Multistream Pipelines

- Implement primary pipeline; place a label on every stage with multiple input or output streams
- Use the endchar “?” to indicate the end of the primary pipeline
- Write the next pipeline, using the labels to refer to streams from the primary pipeline

CMS Pipelines

```
Pipe < INPUT FILE A
| d:drop 4   % label data with dropped data ‘‘d’
| sort 34-36 % sort primary stream
| i:faninany % merge with input ‘‘i’’
| > OUTPUT FILE A
?            % end of primary pipeline
d: | i:      % connect ‘‘d’’ to ‘‘i’’
```

Pipeline Stalls

- Every stage is waiting for some other stage to perform some function (read or write)
- Cause is usually stage that reads multiple inputs in a particular order (or multiple records)
- Preceding stages may not be able to deliver order or quantity required

When a stall occurs, you receive a return code of “-4095”.

Limitations of CMS Pipeines

- Sequential execution on one CPU, no parallelism
- Only available on CMS and z/OS
- Record-oriented

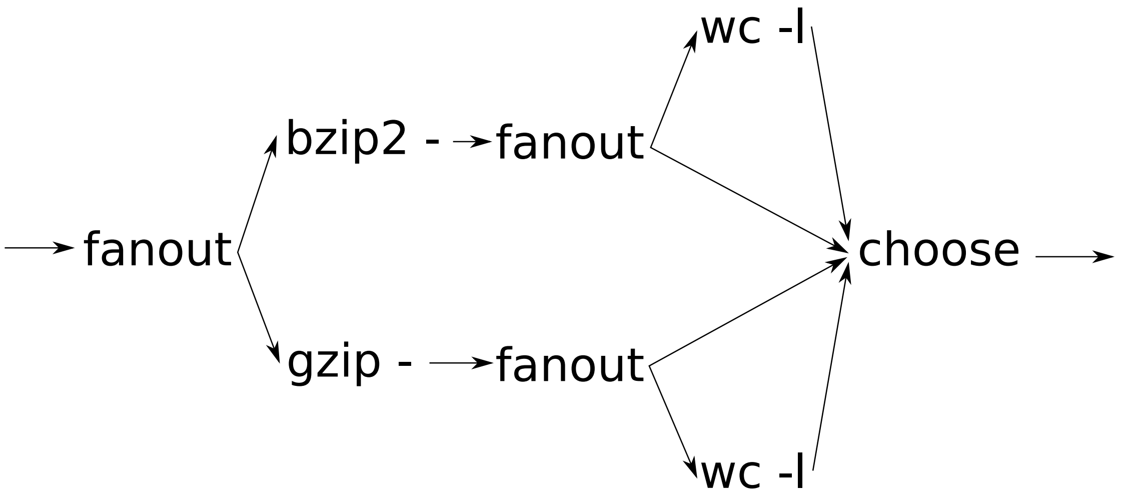
... but these are easy to address!

Our Solution:
DUP $\equiv$ Distributed Multi-Stream Pipelines

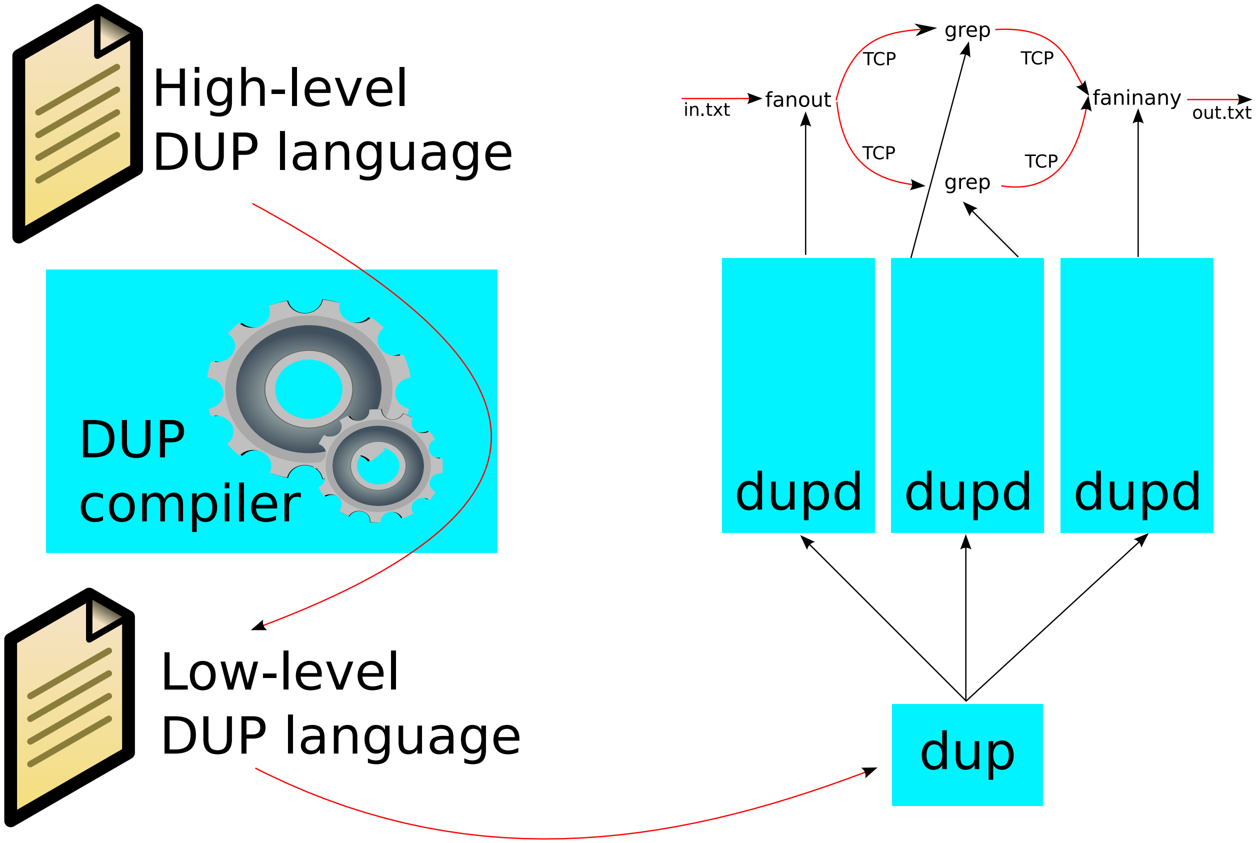
- Computation composed of stages in a flow-graph
- All stages run as individual processes in parallel
- Stages are like UNIX filters, except with possibly multiple inputs and outputs
- DUP used to connect stages
- DUP provides stages for common problems

$\Rightarrow$ Eliminates common problems with parallel programming and guides developers towards modular design

DUP Example



DUP Architecture



High-level DUP language

DUP compiler

Low-level DUP language

DUP Limitations

- Stages communicate via streams

⇒ Computation must be stream-oriented

- Stages run in parallel, internals are up to the stage

⇒ DUP only helps with parallelism if there are enough stages

DUP Application Domains

- Intrusion Detection (sensors, summarization, result distribution)
- Video conferencing
- Event surveillance
- Discrete event simulation
- ...

Future Work

- Develop filters/stages and applications
- High-level DUP programming language (an aspect-oriented coordination mini-language)
- IDE support
- Type systems for streams
- ...

Questions

?