

Performance Counter Monitoring for the Blue Gene/Q Architecture

Heike Jagode

Innovative Computing Laboratory
University of Tennessee, Knoxville

<http://icl.eecs.utk.edu/papi/>

ESP Code for “Q” Workshop
Argonne National Laboratory
April 30 – May 2, 2012

Overview

1. Introduction

2. PAPI overview

3. PAPI for BG/Q

- Processor Unit (PUnit) Component
- L2 Unit Component
- I/O Unit Component
- Network Component
- Compute Node Kernel Unit (CNKUnit) Component

4. Example: 3D-FFT on Q

Introduction

- Very little effort was put into HW performance monitoring tools for the BG/Q predecessor BG/P
- HPC community was left behind with rather poor and incomplete methods
- To eliminate this limitation, for BG/Q we planned carefully and collaborate closely with IBM's Performance Group

Result:

- Added 5 new Components to PAPI to support HW performance monitoring for the BG/Q network, the I/O system, the Compute Node Kernel in addition to the processing cores

PAPI

- **Middleware** that provides a consistent and efficient programming interface for the performance counter hardware found in most major microprocessors.
- Started as a Parallel Tools Consortium project in 1998
 - Goal was to produce a specification for a **portable interface** to the hardware performance counters.
- Countable events are defined in two ways:
 - Platform-neutral **Preset Events** (e.g., PAPI_TOT_INS)
 - Platform-dependent **Native Events** (e.g., L3_CACHE_MISS)
- Preset Events can be **derived** from multiple Native Events (e.g. PAPI_L1_TCM might be the sum of L1 Data Misses and L1 Instruction Misses on a given platform)

PAPI Hardware Events

Preset Events

- Standard set of over 100 events for application performance tuning
- No standardization of the exact definition
- Mapped to either single or linear combinations of native events on each platform
- Use *papi_avail* to see what preset events are available on a given platform

Native Events

- Any event countable by the CPU
- Same interface as for preset events
- Use *papi_native_avail* utility to see all available native events

Use *papi_event_chooser* utility to select a compatible set of events

Overview

1. Introduction

2. PAPI overview

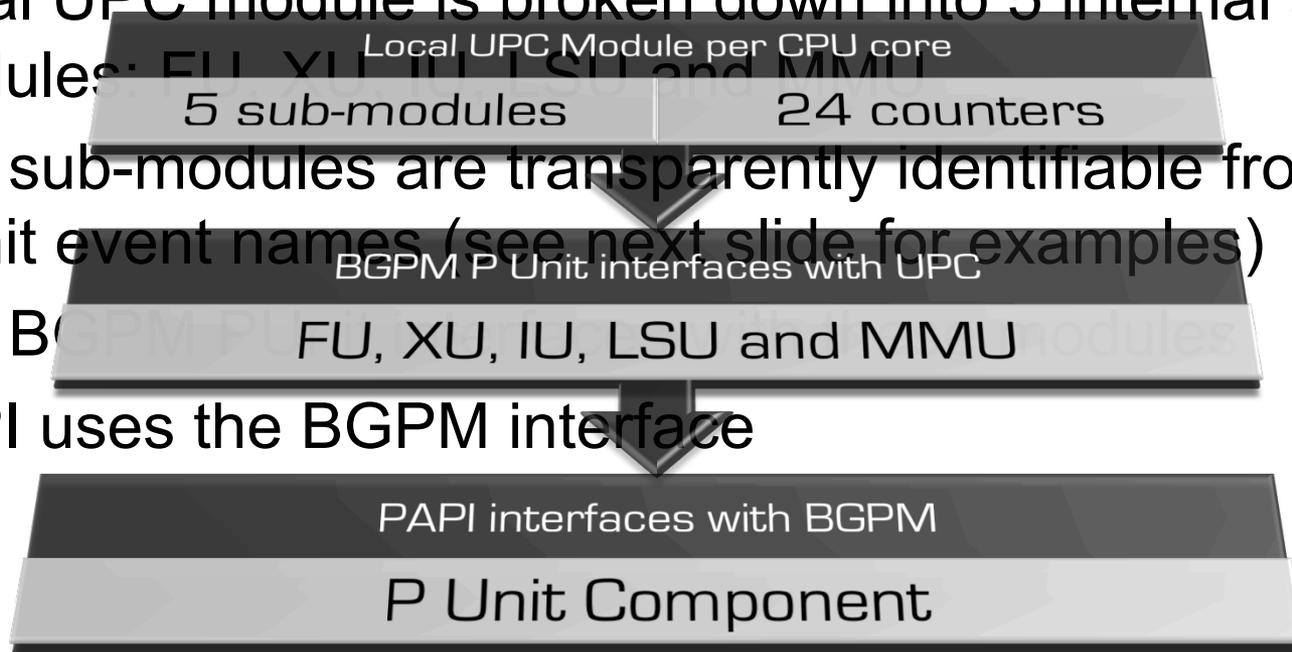
3. PAPI for BG/Q

- Processor Unit (PUnit) Component
- L2 Unit Component
- I/O Unit Component
- Network Component
- Compute Node Kernel Unit (CNKUnit) Component

4. Example: 3D-FFT on Q

PUnit Component

- Each of the 18 A2 CPU cores has a local UPC module
 - Each of these modules provides 24 counters (14-bit) to sample A2 events, L1 cache related events, floating point operations, etc.
 - Local UPC module is broken down into 5 internal sub-modules: FU, XU, IU, LSU and MMU
 - The sub-modules are transparently identifiable from the PUnit event names (see next slide for examples)
- The BGPM P Unit interfaces with UPC
- PAPI uses the BGPM interface



PUnit Events (Native | Presets)

- Currently, there are 269 native PUnit events available:

PUnit Event	Description
PEVT_AXU_INSTR_COMMIT	A valid AXU (non-load/store) instruction is in EX6, past the last flush point. - AXU uCode sub-operations are also counted by PEVT_XU_COMMIT instead.
PEVT_AXU_CR_COMMIT	A valid AXU CR updater instruction is in EX6, past the last flush point.
PEVT_AXU_IDLE	No valid AXU instruction is in the EX6 stage.
...	...
PEVT_IU_IL1_MISS	A thread is waiting for a reload from the L2. - Not when CI=1. - Not when thread held off for a reload that another thread is waiting for. - Still counts even if flush has occurred.
PEVT_IU_IL1_MISS_CYC	Number of cycles a thread is waiting for a reload from the L2. - Not when CI=1. - Not when thread held off for a reload that another thread is waiting for. - Still counts even if flush has occurred.
PEVT_IU_IL1_RELOADS_DROPPED	Number of times a reload from the L2 is dropped, per thread - Not when CI=1 - Does not count when not loading cache due to a back invalidate to that address
...	...
PEVT_XU_BR_COMMIT_CORE	Number of Branches committed
PEVT_XU_BR_MISPRED_COMMIT_CORE	Number of mispredicted Branches committed (does not include target address mispredicted)

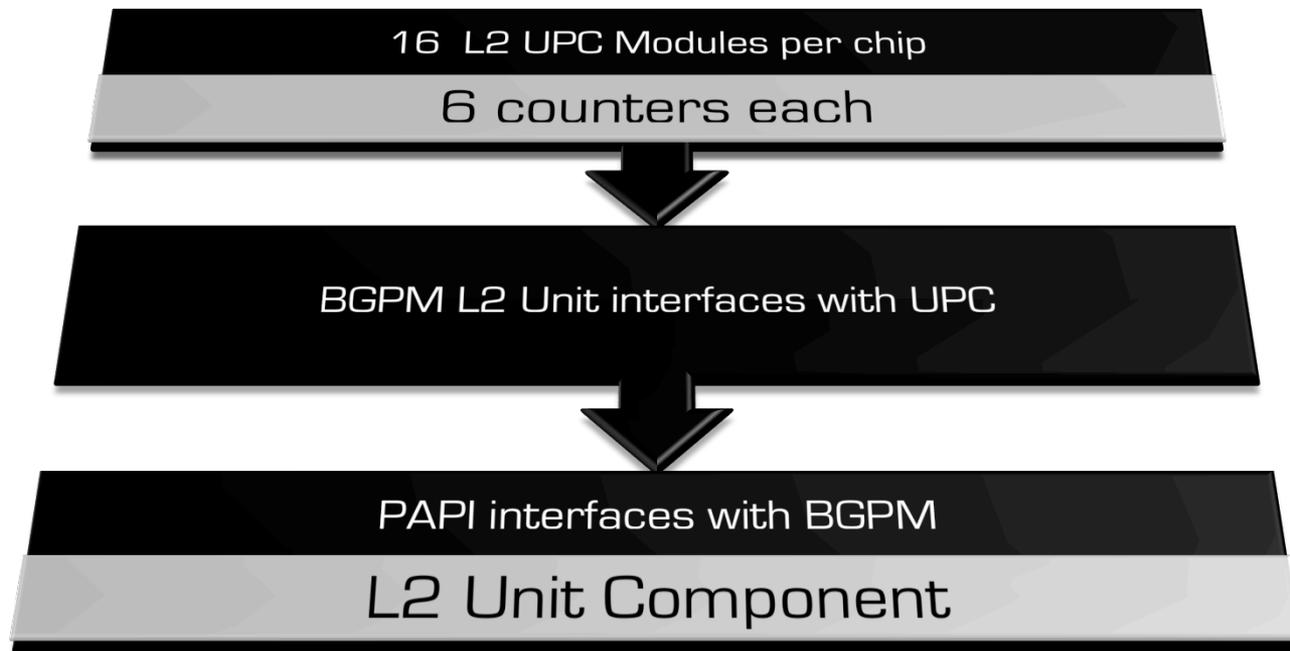
PUnit Events (Native | Presets)

- Currently, there are 269 native PUnit events available:
- Out of 107 possible predefined events, there are currently 41 events available of which 12 are derived events:

Name	Code	Avail	Deriv	Description (Note)
PAPI_L1_ICM	0x80000001	Yes	No	Level 1 instruction cache misses
PAPI_FXU_IDL	0x80000011	Yes	No	Cycles integer units are idle
PAPI_TLB_DM	0x80000014	Yes	Yes	Data translation lookaside buffer misses
PAPI_TLB_IM	0x80000015	Yes	No	Instruction translation lookaside buffer misses
PAPI_TLB_TL	0x80000016	Yes	Yes	Total translation lookaside buffer misses
PAPI_L1_LDM	0x80000017	Yes	No	Level 1 load misses
PAPI_L1_STM	0x80000018	Yes	No	Level 1 store misses
PAPI_BTAC_M	0x8000001b	Yes	No	Branch target address cache misses
PAPI_PRF_DM	0x8000001c	Yes	No	Data prefetch cache misses
PAPI_TLB_SD	0x8000001e	Yes	No	Translation lookaside buffer shutdowns
PAPI_CSR_FAL	0x8000001f	Yes	No	Failed store conditional instructions
PAPI_CSR_SU	0x80000020	Yes	Yes	Successful store conditional instructions
PAPI_CSR_TOT	0x80000021	Yes	No	Total store conditional instructions

L2 Unit Component

- Shared L2 cache is split into 16 separate slices
- Each of the 16 L2 memory slices has a L2 UPC module that provides 6 counters (node-wide)



L2 Unit Native Events

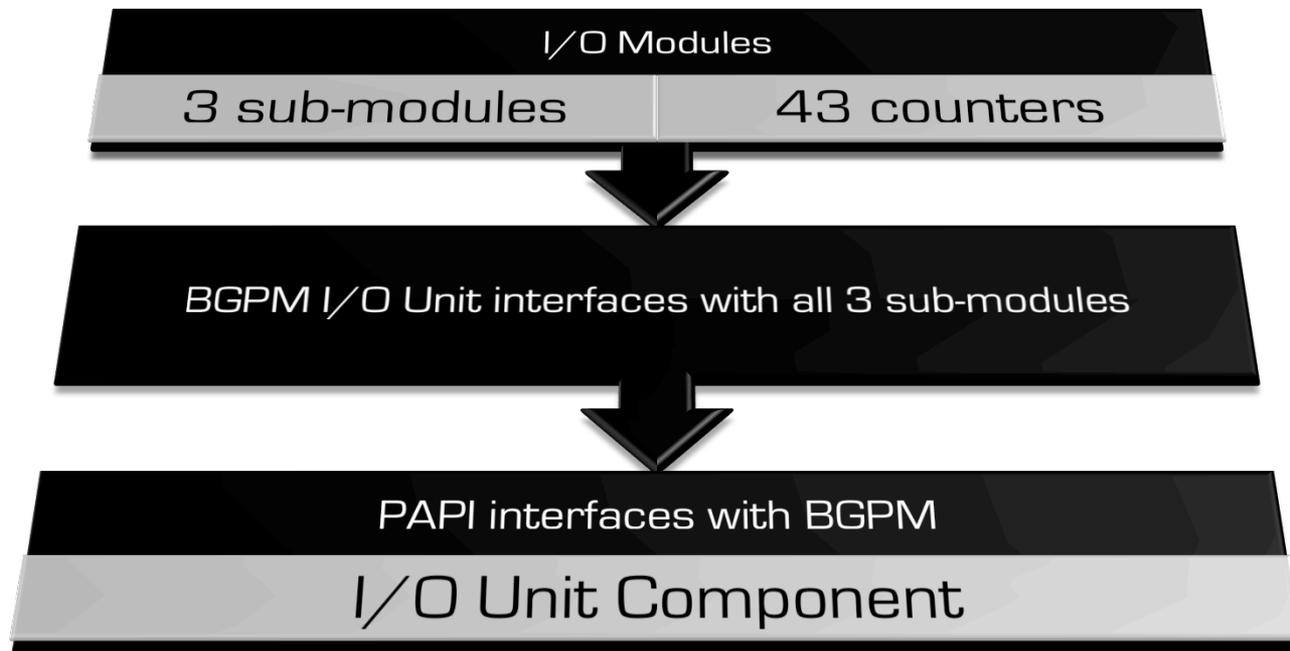
- Currently, there are 32 L2 Unit events available:

L2Unit Event	Description
PEVT_L2_HITS	hits in L2, both load and store. Network Polling store operations from core 17 on BG/Q pollute in this count during normal use
PEVT_L2_MISSES	cacheline miss in L2 (both loads and stores)
PEVT_L2_PREFETCH	fetching cacheline ahead of L1P prefetch
...	...

- BG/Q processor has two DDR3 memory controllers, each
- interfacing with eight slices of the L2 cache to handle their cache misses (one controllers for each half of the 16 cores on the chip)
- The counting hardware can either keep the counts from each slice separate, or combine the counts from each slice into single values (default)

I/O Unit Component

- The Message, PCIe, and DevBus module – which are collectively referred to as I/O modules – provide together 43 counters (node-wide)



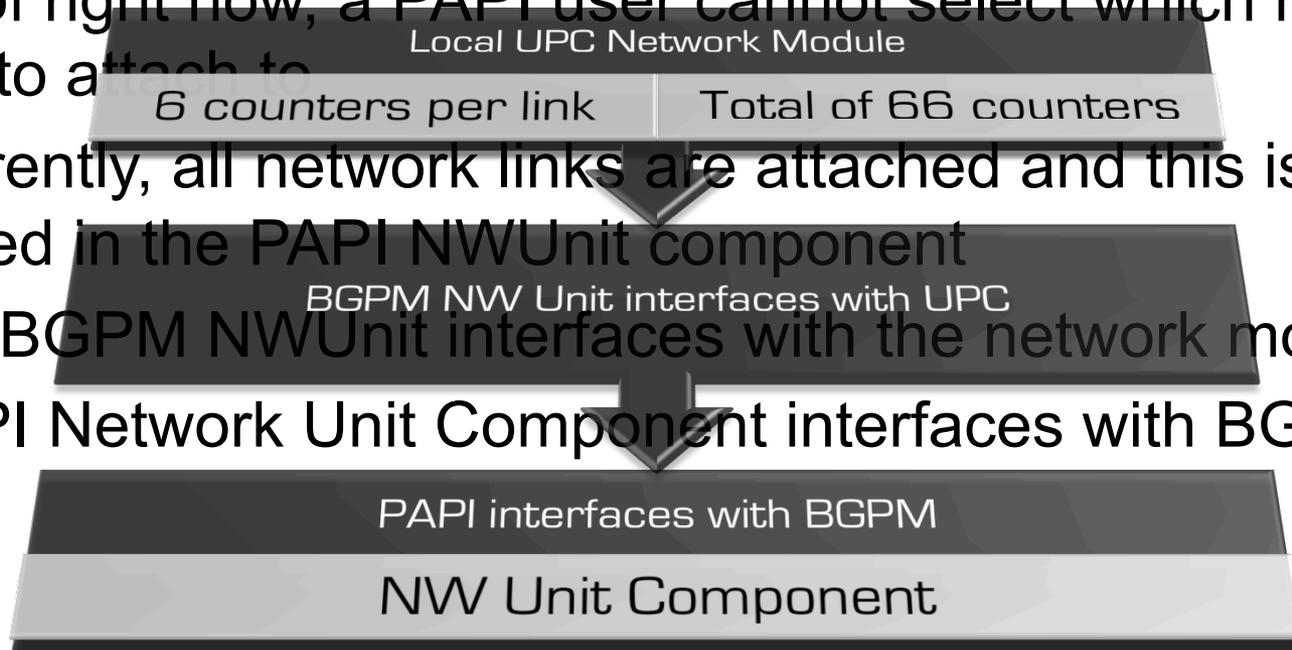
I/O Unit Native Events

- Currently, there are 44 I/O Unit events available
- The three I/O sub-modules are transparently identifiable from the I/O Unit event names

IOUnit Event	Description
PEVT_MU_PKT_INJ	A new packet has been injected (Packet has been stored to ND FIFO)
PEVT_MU_MSG_INJ	A new message has been injected (All packets of the message have been stored to ND FIFO)
PEVT_MU_FIFO_PKT_RCV	A new FIFO packet has been received (The packet has been stored to L2. There is no pending switch request)
...	...
PEVT_PCIE_INB_RD_BYTES	Inbound Read Bytes Request
PEVT_PCIE_INB_RDS	Inbound Read Request
PEVT_PCIE_INB_RD_CMPLT	Inbound Read Completion
...	...
PEVT_DB_PCIE_INB_WRT_BYTES	PCIe inbound write bytes written
PEVT_DB_PCIE_OUTB_RD_BYTES	PCIe outbound read bytes requested
PEVT_DB_PCIE_OUTB_RDS	PCIe outbound read request
...	...

Network Unit Component

- The 5D-Torus network provides a local UPC network module with 66 counters - each of the 11 links has six 64-bit counters
 - As of right now, a PAPI user cannot select which network link to attach to
 - Currently, all network links are attached and this is hard-coded in the PAPI NWUnit component
- The BGPM NWUnit interfaces with the network modules
- PAPI Network Unit Component interfaces with BGPM



Network Unit Native Events

- Currently, there are 31 Network Unit events available

NWUnit Event	Description
PEVT_NW_USER_PP_SENT	Number of 32 byte user point to point packet chunks sent. Includes packets originating or passing through the current node
PEVT_NW_USER_DYN_PP_SENT	Number of 32 byte user dynamic point to point packet chunks sent. Includes packets originating or passing through the current node
PEVT_NW_USER_ESC_PP_SENT	Number of 32 byte user escape point to point packet chunks sent. Includes packets originating or passing through the current node
...	...

CNK Unit Component

- CNK is the lightweight Compute Node Kernel that runs on all the 16 compute cores
- BGPM offers a “virtual” CNK Unit that has software counters collected by the kernel (kernel counter values are read via a system call)
- Currently, there are 29 CNK Unit events available

CNKUnit Event	Description
PEVT_CNKHWT_SYSCALL	External Input Interrupt
PEVT_CNKHWT_CRITICAL	Critical Input Interrupt
PEVT_CNKHWT_FIT	Fixed Interval Timer Interrupt
...	...

Overflow and Multiplexing

Overflow:

- Only the local UPC module, L2 and I/O UPC hardware support performance monitor interrupts when a programmed counter overflows
- For that reason, only the PUnit, L2Unit, and I/OUnit provide overflow support in BGPM and PAPI

Multiplexing:

- PAPI supports multiplexing for the BG/Q platform
- The BGPM PUnit does not directly implement multiplexing of event sets; but, it does indirectly support multiplexing by supporting a multiplexed event set type

Overview

1. Introduction

2. PAPI overview

3. PAPI for BG/Q

- Processor Unit (PUnit) Component
- L2 Unit Component
- I/O Unit Component
- Network Component
- Compute Node Kernel Unit (CNKUnit) Component

4. Example: 3D-FFT on Q

BG/Q network

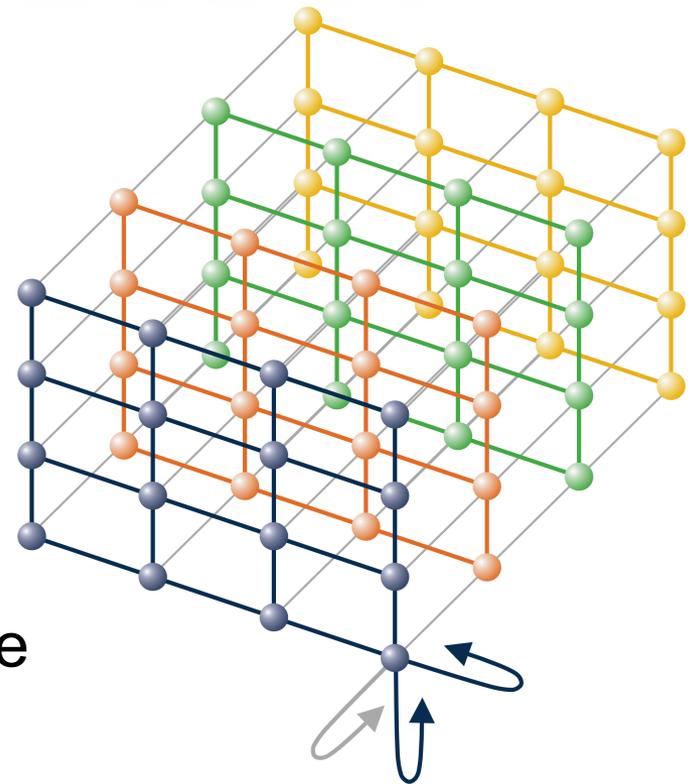
BG/L+P:

Compute nodes organized as a
3D-torus

MAIN FEATURE:

every node is connected to
its **six** neighbour nodes through
bidirectional links

To maintain application performance,
correct mapping of MPI tasks onto the
torus network is a critical factor



BG/Q network

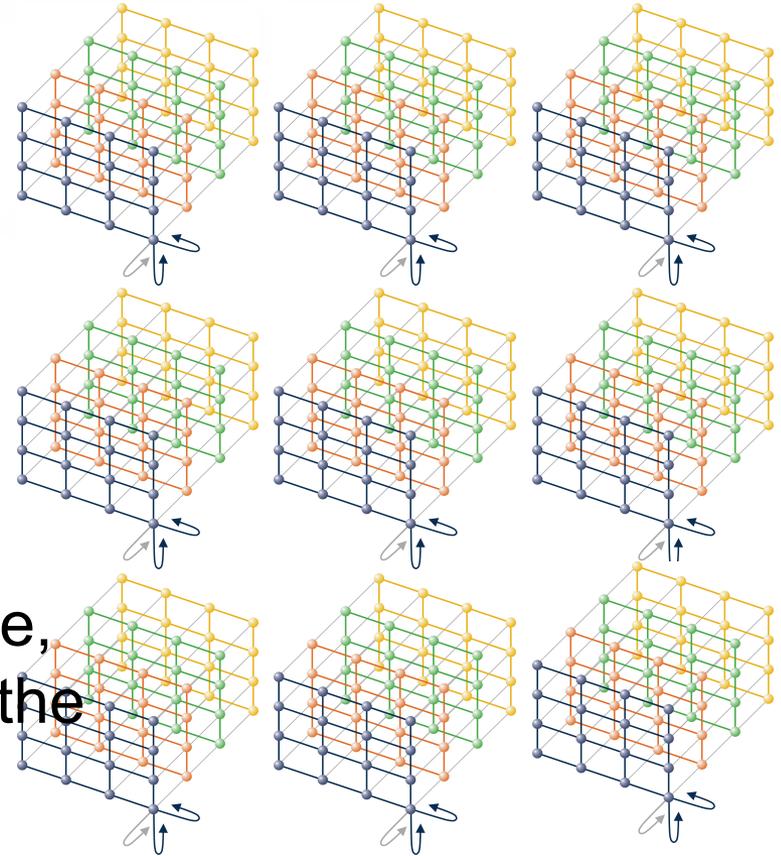
BG/Q:

Compute nodes organized as a **5D-torus**

MAIN FEATURE:

every node is connected to its **ten** neighbour nodes through bidirectional links

To maintain application performance, correct mapping of MPI tasks onto the **torus** network is a critical factor



3D FFT on BG/Q

- Why multi-dimensional FFT ?
- Computation performed in three single stages:

$$A_{x,y,z} \in \mathbb{C} \quad x, y, z \in \mathbb{Z} \quad \begin{array}{l} \forall x, 0 \leq x < L \\ \forall y, 0 \leq y < M \\ \forall z, 0 \leq z < N \end{array}$$

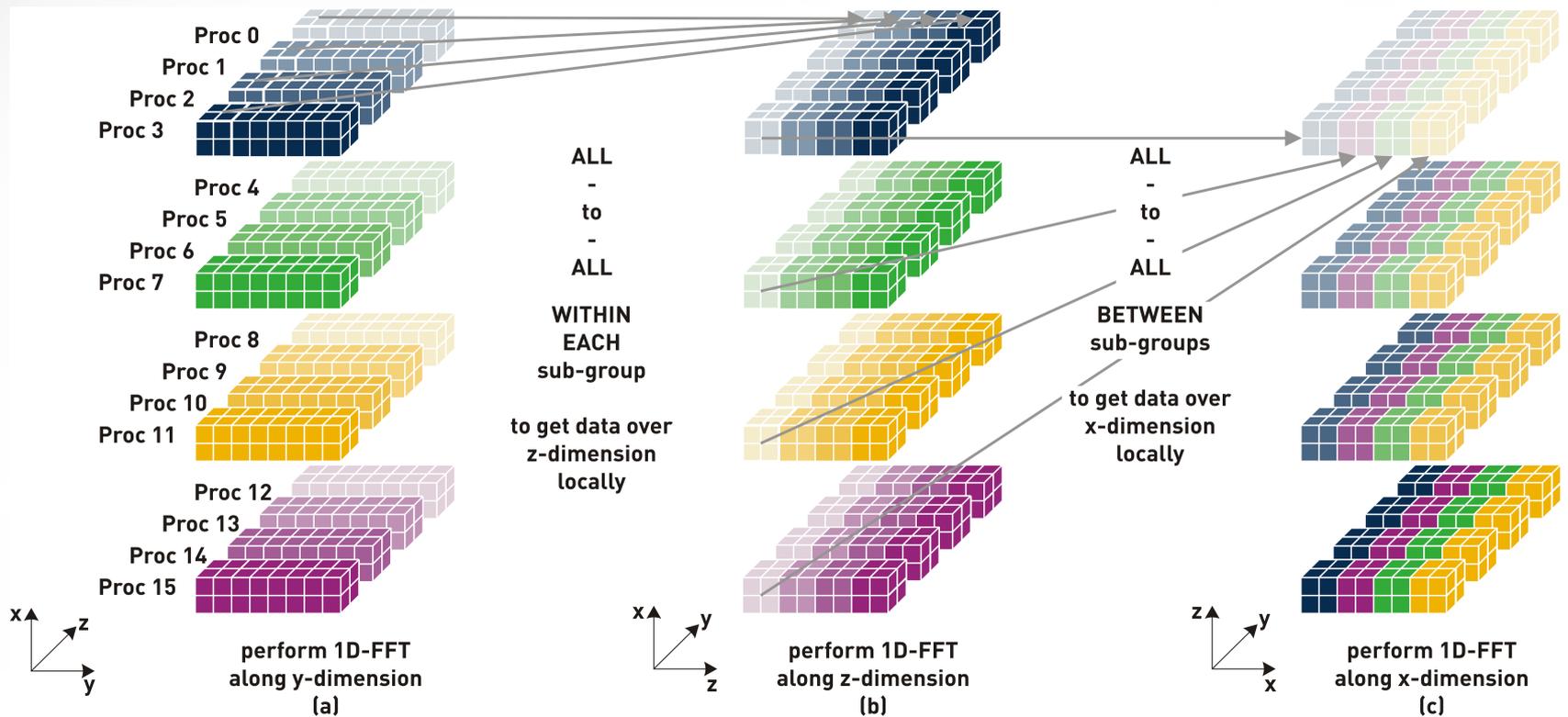
$$\tilde{A}_{u,v,w} := \underbrace{\sum_{x=0}^{L-1} \sum_{z=0}^{N-1} \sum_{y=0}^{M-1} A_{x,y,z} \exp\left(-2\pi i \frac{vy}{M}\right)}_{\text{1st 1D FT along } y} \exp\left(-2\pi i \frac{wz}{N}\right) \exp\left(-2\pi i \frac{ux}{L}\right)$$

2nd 1D FT along z

3rd 1D FT along x

2D Decomposition

MPI tasks organized in 2D virtual processor grid using MPI Cartesian grid topology construct



Example: Low Level API

```
#include "papi.h"
#define NUM_EVENTS 2
char events[NUM_EVENTS] = { "PEVT_NW_USER_PP_SENT",
                            "PEVT_NW_USER_DYN_PP_SENT" };

int EventSet = PAPI_NULL;
long long values[NUM_EVENTS];

/* Initialize the Library */
retval = PAPI_library_init ( PAPI_VER_CURRENT );
/* convert native events to PAPI code */
for( h = 0; h < NUM_EVENTS; h++ )
    retval = PAPI_event_name_to_code( EventName[h], &events[h] );
/* Allocate space for the new EventSet and do setup */
retval = PAPI_create_eventset ( &EventSet );
/* Add Flops and total cycles to the eventset */
retval = PAPI_add_events ( EventSet, events, NUM_EVENTS );

/* Start the counters */
retval = PAPI_start ( EventSet );

do_work(); /* What we want to monitor: MPI_Alltoall( .. ) */

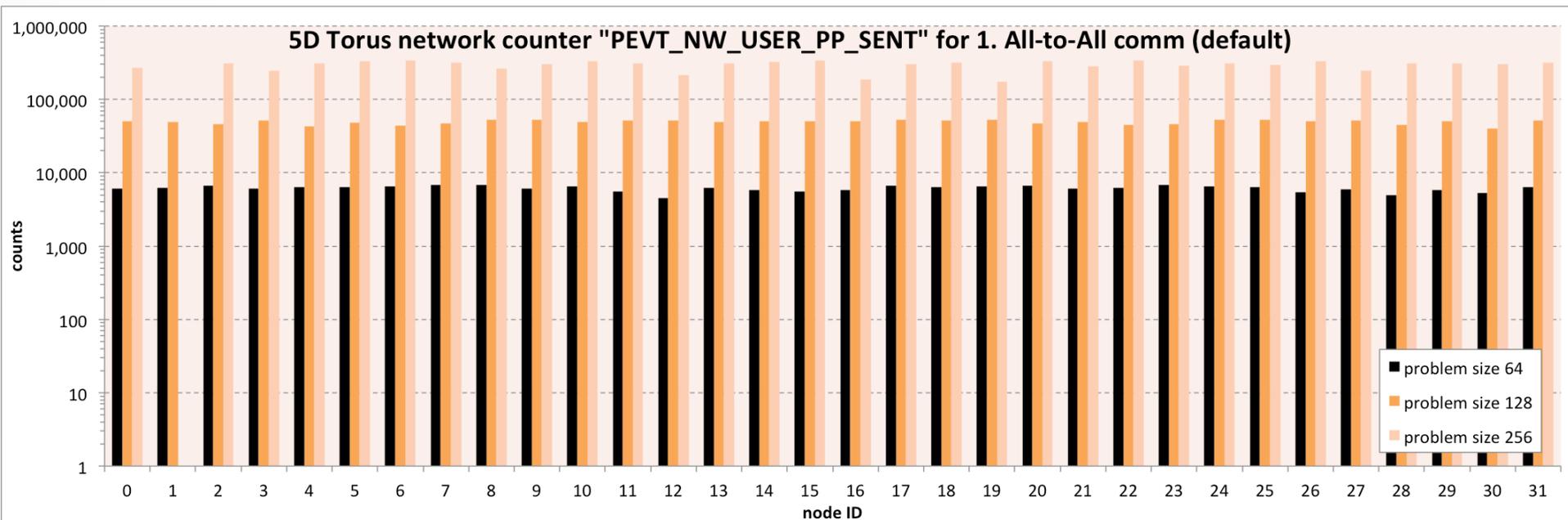
/*Stop counters and store results in values */
retval = PAPI_stop ( EventSet, values );
```

PAPI measurements

- 32 nodes:

	A	B	C	D	E	T
nodes	2	2	2	2	2	16
torus	0	0	0	0	1	-

- 512 MPI tasks, 2D virtual processor grid: { 8, 64 }
- "PEVT_NW_USER_PP_SENT": # of 32B user p2p packet chunks sent. Includes packets originating or passing through the current node.



PAPI measurements

- 32 nodes:

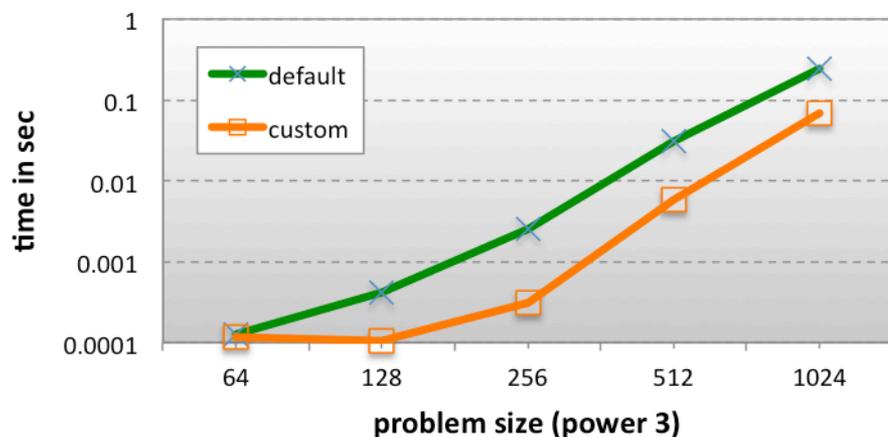
	A	B	C	D	E	T
nodes	2	2	2	2	2	16
torus	0	0	0	0	1	-

- 512 MPI tasks, 2D virtual processor grid: { 8, 64 }
- “PEVT_NW_USER_PP_SENT”: # of 32B user p2p packet chunks sent. Includes packets originating or passing through the current node.



Results

1. All-to-All



Comm	8%	4x	8.4x	5.4x	3.6x
------	----	----	------	------	------

3D-FFT	9%	22%	12%	6%	6%
--------	----	-----	-----	----	----

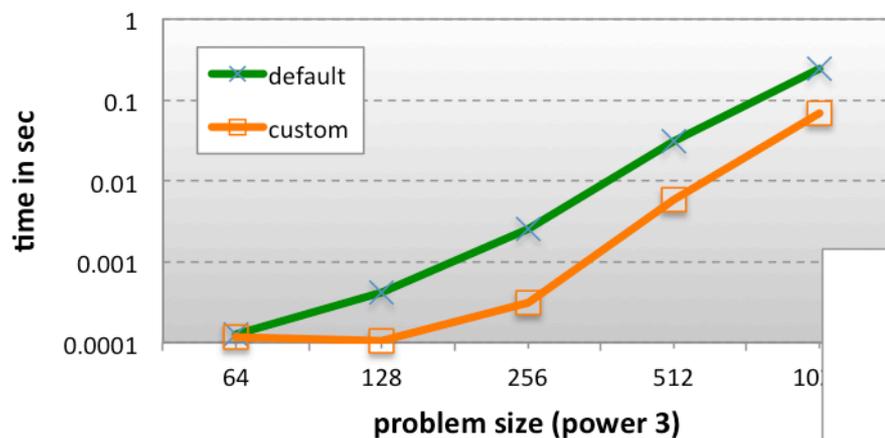
rank	A	B	C	D	E	T
0	0	0	0	0	0	0
64	0	0	1	0	0	0
128	0	1	0	0	0	0
192	0	1	1	0	0	0
256	1	0	0	0	0	0
320	1	0	1	0	0	0
384	1	1	0	0	0	0
448	1	1	1	0	0	0

rank	A	B	C	D	E	T
0	0	0	0	0	0	0
64	0	0	0	0	0	1
128	0	0	0	0	0	2
192	0	0	0	0	0	3
256	0	0	0	0	0	4
320	0	0	0	0	0	5
384	0	0	0	0	0	6
448	0	0	0	0	0	7

Results

rank	A	B	C	D	E	T
0	0	0	0	0	0	0
64	0	0	1	0	0	0
128	0	1	0	0	0	0
192	0	1	1	0	0	0
256	1	0	0	0	0	0
320	1	0	1	0	0	0
384	1	1	0	0	0	0
448	1	1	1	0	0	0

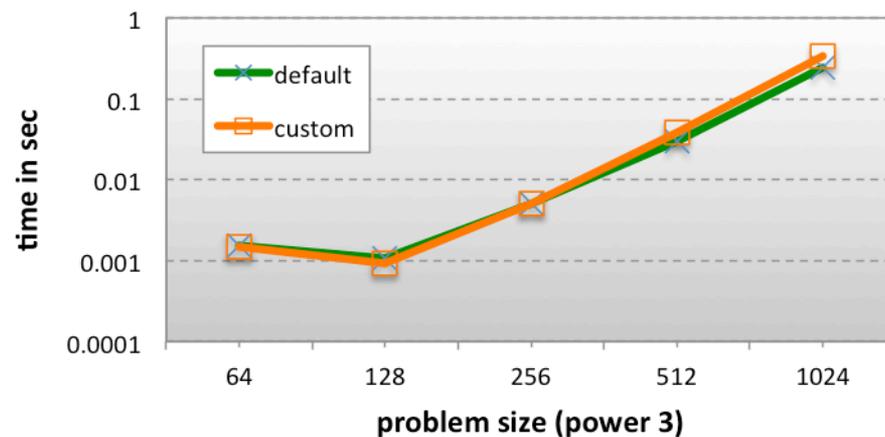
1. All-to-All



Comm 8% 4x 8.4x 5.4x 3

3D-FFT 9% 22% 12% 6%

2. All-to-All



Summary

- Performance analysis tools for parallel applications running on large scale systems rely on HW perf counters to gather perf relevant data from the system

PAPI's 5 new components for BG/Q

- Enable HW perf counter monitoring for
 - Processing unit
 - 5D Torus
 - I/O system
 - Compute Node Kernel

Very Early Access Validation Example:

- 3D-FFT kernel – instrumented with PAPI – for comm evaluation
- 5D torus network counters detect tons of inter-node communications that were redundant

PAPI on VEAS / CETUS

- Installed in `/soft/perftools/papi`
- Utilities in `bin` directory
 - `papi_avail` , `papi_native_avail` , etc.
 - Run on compute node using `qsub` – e.g.,
`qsub -n 1 --mode c1 -t 10 papi_avail`
- Examples in `/home/jagode/public`
 - `cd /home/jagode/public`
 - `cp -r papi <your_choice>`
 - `cd papi/src/ctests`
 - Run on compute node using `qsub` – e.g.,
`qsub -n 1 --mode c1 -t 10 first`

Acknowledgement

The general availability of PAPI for BG/Q – that can be utilized *immediately* by end users – is due to a cooperative effort of several parties.

A special acknowledgment goes to the **IBM performance team**, especially
Roy Musselman
Kris Davis

for the careful planning long before the BG/Q release as well as the close partnership and joint effort.